

Zhong-Zhi Shi  
Ramakoti Sadananda (Eds.)

LNAI 4088

# Agent Computing and Multi-Agent Systems

9th Pacific Rim International Workshop  
on Multi-Agents, PRIMA 2006  
Guilin, China, August 2006, Proceedings

 Springer

Lecture Notes in Artificial Intelligence 4088

Edited by J. G. Carbonell and J. Siekmann

Subseries of Lecture Notes in Computer Science

Zhong-Zhi Shi Ramakoti Sadananda (Eds.)

# Agent Computing and Multi-Agent Systems

9th Pacific Rim International Workshop  
on Multi-Agents, PRIMA 2006  
Guilin, China, August 7-8, 2006  
Proceedings

Series Editors

Jaime G. Carbonell, Carnegie Mellon University, Pittsburgh, PA, USA  
Jörg Siekmann, University of Saarland, Saarbrücken, Germany

Volume Editors

Zhong-Zhi Shi  
Chinese Academy of Sciences  
Institute of Computing Technology  
Beijing 100080, China  
E-mail: shizz@ics.ict.ac.cn

Ramakoti Sadananda  
Asian Institute of Technology  
P.O. Box 4, Klong Luang, Pathumthani 12120, Thailand  
E-mail: sada@cs.ait.ac.th

Library of Congress Control Number: 2006929806

CR Subject Classification (1998): I.2.11, I.2, C.2.4, D.2, F.3

LNCS Sublibrary: SL 7 – Artificial Intelligence

ISSN 0302-9743  
ISBN-10 3-540-36707-1 Springer Berlin Heidelberg New York  
ISBN-13 978-3-540-36707-9 Springer Berlin Heidelberg New York

This work is subject to copyright. All rights are reserved, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, re-use of illustrations, recitation, broadcasting, reproduction on microfilms or in any other way, and storage in data banks. Duplication of this publication or parts thereof is permitted only under the provisions of the German Copyright Law of September 9, 1965, in its current version, and permission for use must always be obtained from Springer. Violations are liable to prosecution under the German Copyright Law.

Springer is a part of Springer Science+Business Media  
springer.com

© Springer-Verlag Berlin Heidelberg 2006  
Printed in Germany

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India  
Printed on acid-free paper SPIN: 11802372 06/3142 5 4 3 2 1 0



## Preface

PRIMA is a series of workshops on agent computing and multi-agent systems, integrating the activities in Asia and Pacific Rim countries. Agent computing and multi-agent systems are computational systems in which several autonomous or semi-autonomous agents interact with each other or work together to perform some set of tasks or satisfy some set of goals. These systems may involve computational agents that are homogeneous or heterogeneous, they may involve activities on the part of agents having common or distinct goals, and they may involve participation on the part of humans and intelligent agents.

The aim of PRIMA 2006 was to bring together Asian and Pacific Rim researchers and developers from academia and industry to report on the latest technical advances or domain applications and to discuss and explore scientific and practical problems as raised by the participants.

PRIMA 2006 received 203 submitted papers. Each paper was reviewed by two internationally renowned Program Committee members. After careful reviews, 39 regular papers and 57 short papers were selected for this volume. We would like to thank all the authors who submitted papers to the workshop. We are very grateful to all Program Committee members and reviewers for their splendid work in reviewing the papers. Finally, we thank the editorial staff of Springer for publishing this volume in the *Lecture Notes in Artificial Intelligence* series.

For more information about PRIMA, please visit the following websites:

PRIMA <http://www.ai.soc.i.kyoto-u.ac.jp/prima/>

PRIMA 1998: <http://www.ai.soc.i.kyoto-u.ac.jp/prima98/>

PRIMA 1999: [www.ai.soc.i.kyoto-u.ac.jp/prima99/](http://www.ai.soc.i.kyoto-u.ac.jp/prima99/)

PRIMA 2000: <http://www.ai.soc.i.kyoto-u.ac.jp/prima2000/>

PRIMA 2001: <http://www.ai.soc.i.kyoto-u.ac.jp/prima2001/>

PRIMA 2002: <http://www.ai.soc.i.kyoto-u.ac.jp/prima2001/>

PRIMA 2003: <http://prima.uos.ac.kr/>

PRIMA 2004: <https://www.cs.auckland.ac.nz/prima04/>

PRIMA 2005: <http://www.prima2005.org/home.htm>

PRIMA 2006: <http://www.intsci.ac.cn/PRIMA2006/index.jsp>

August 2006

Zhongzhi Shi  
Ramakoti Sadananda

# Organization

## Conference Committee

General Chair	Toru Ishida (Japan)
Program Chairs	Zhongzhi Shi (China)
	Ramakoti Sadananda (Thailand)

## Program Committee

Mohd Sharifuddin Ahmad (Malaysia)	Mike Barley (New Zealand)
Penny Baillie-de Byl (Australia)	Joongmin Choi (Korea)
Stephen Cranefield (New Zealand)	Jirapun Daengdej (Thailand)
Jingbo Dang (USA)	John Debenham (Australia)
Klaus Fisher (Germany)	Zili Zhang (Australia)
Yanxiang He (China)	Bryan Horling (USA)
Chun-Nan Hsu (Taiwan)	Jun Hu (China)
Shanli Hu (China)	Michael Huhns (USA)
Toru Ishida (Japan)	Ilkon Kim (Korea)
Incheol Kim (Korea)	Yasuhiko Kitamura (Japan)
Jean-Luc Koning (France)	Robert Kremer (Canada)
Kazuhiro Kuwabara (Japan)	Jaeho Lee (Korea)
Jimmy H.M. Lee (China)	Ho-fung Leung (China)
Wei Li (Australia)	Yongquan Liang (China)
Lejian Liao (China)	Chao-Lin Liu (Taiwan)
Jiming Liu (China)	Jyi-Shane Liu (Taiwan)
Rey-Long Liu (Taiwan)	Jian Lv (China)
Dickson Lukose (Malaysia)	Xudong Luo (UK)
Joerg Mueller (Germany)	Shivashankar B. Nair (India)
Sascha Ossowski (Spain)	Somnuk Phon-Amnuaisuk (Malaysia)
Yuhui Qiu (China)	Anita Raja (USA)
Ali Selamat (Malaysia)	Raymund Sison (Philippines)
Von-Wun Soo (Taiwan)	Toshiharu Sugawara (Japan)
Jung-Jin Yang (Korea)	Soe-Tsyr Yuan (Taiwan)
Laura Zavala (USA)	Minjie Zhang (Australia)
Shensheng Zhang (China)	

## Referees

Joongmin Choi	Stephen Cranefield	Zili Zhang
Yanxiang He	Chun-Nan Hsu	Shanli Hu
Yasuhiko Kitamura	Kazuhiro Kuwabara	Jaeho Lee

VIII Organization

Jimmy H.M. Lee  
Lejian Liao  
Jyi-Shane Liu  
Ali Selamat  
Soe-Tsyr Yuan  
Xiaoyun Chen  
Changjie Wang  
Chi-kong Chan  
Zhu Liehuang  
Alberto Fernandez  
Bei-shui Liao  
Yinglong Ma  
Wei Wang  
Zheng Zhang  
Qingyong Li  
Gang Chen  
Zhiping Shi

Ho-fung Leung  
Chao-Lin Liu  
Xudong Luo  
Toshiharu Sugawara  
Minjie Zhang  
Huiye Ma  
Ching-man Au Yeung  
Dickson K. W. Chiu  
Cesar Caceres  
Ruben Ortiz  
Zining Cao  
Seungkeun Lee  
Deguo Yang  
He Huang  
Jiewen Luo  
Lirong Qiu  
Zhiyong Zhang

Wei Li  
Rey-Long Liu  
Sascha Ossowski  
Jung-Jin Yang  
Genjian Yu  
Ka-man Lam  
Maria S. L. Lin  
Zhao Qingjie  
Holger Billhardt  
Sergio Saugar  
Jun Hu  
Hui Wang  
Young Ik Eom  
Rui Huang  
Ping Luo  
Chuan Shi  
Maoguang Wang

# Table of Contents

## Invited Talks

Agent and Grid Technologies for Intercultural Collaboration <i>Toru Ishida</i> .....	1
Agent Grid Collaborative Environment <i>Zhongzhi Shi</i> .....	5
An Agent-Based System Integration Architecture for Intelligent Service Robots <i>Jaeho Lee</i> .....	6
DartGrid: A Semantic Grid and Application for Traditional Chinese Medicine <i>Zhaohui Wu</i> .....	7

## Agent Model

A Grammatical Framework for Modelling Multi-agent Dialogues <i>Gemma Bel-Enguix, María Adela Grando, M. Dolores Jiménez-López</i> .....	10
A Calculus for MAS Interaction Protocol <i>Hongbing Chen, Qun Yang, Manwu Xu</i> .....	22
Synthesizing Stigmergy for Multi Agent Systems <i>Grant Blaise O'Reilly, Elizabeth Ehlers</i> .....	34
Model Checking for Epistemic and Temporal Properties of Uncertain Agents <i>Zining Cao</i> .....	46

## Agent Architectures

A Task Management Architecture for Control of Intelligent Robots <i>Jaeho Lee, Byulsaim Kwak</i> .....	59
Multi-agent Based Selfish Routing for Multi-channel Wireless Mesh Networks <i>Yanxiang He, Jun Xiao</i> .....	71

Natural Language Communication Between Human and Artificial Agents  
*Christel Kemke* ..... 84

An Extended BDI Agent with Policies and Contracts  
*Bei-shui Liao, Hua-xin Huang, Ji Gao* ..... 94

**Agent-Oriented Software Engineering**

Towards a Customized Methodology to Develop Multi-Agent Systems  
*Xiao Xue, Xingquan Liu, Rong Li* ..... 105

A Systematic Methodology for Adaptive Systems in Open Environments  
*Li-ming Wang, Ya-chong Li* ..... 117

Multi-modal Services for Web Information Collection Based on Multi-agent Techniques  
*Qing He, Xiurong Zhao, Sulan Zhang* ..... 129

Formalizing Risk Strategies and Risk Strategy Equilibrium in Agent Interactions Modeled as Infinitely Repeated Games  
*Ka-man Lam, Ho-fung Leung* ..... 138

**Agent Grid**

Reverse Auction-Based Grid Resources Allocation  
*Zhengyou Liang, Yu Sun, Ling Zhang, Shoubin Dong* ..... 150

Data Grid System Based on Agent for Interoperability of Distributed Data  
*Youn-Gyou Kook, Gye-Dong Jung, Young-Keun Choi* ..... 162

A Layered Semantics for Mobile Computation  
*Jianghua Lv, Shilong Ma, Jing Pan, Li Ma* ..... 175

Immunity and Mobile Agent Based Intrusion Detection for Grid  
*Xun Gong, Tao Li, Ji Lu, Tiejang Wang, Gang Liang, Jin Yang, Feizian Sun* ..... 187

**Semantic Web Services**

Description Logic Based Composition of Web Services  
*Fen Lin, Lirong Qiu, He Huang, Qing Yu, Zhongzhi Shi* ..... 199

A Reputation Multi-agent System in Semantic Web <i>Wei Wang, Guosun Zeng, Lulai Yuan</i> .....	211
Ontological Modeling of Virtual Organization Agents <i>Lejian Liao, Liehuang Zhu, Jing Qiu</i> .....	220
Parameter Evolution for Quality of Service in Multimedia Networking <i>Ji Lu, Tao Li, Xun Gong</i> .....	233

## Collaboration

A DDL Based Formal Policy Representation <i>Maoguang Wang, Li Zeng, Jiewen Luo, Qing Yu</i> .....	245
Concurrent Agent Social Strategy Diffusion with the Unification Trend <i>Yichuan Jiang, Toru Ishida</i> .....	256
Exploiting Based Pre-testing in Competition Environment <i>Li-ming Wang, Yang Bai</i> .....	269
Teamwork Formation for <i>Keepaway</i> in Robotics Soccer (Reinforcement Learning Approach) <i>Nobuyuki Tanaka, Sachiyo Arai</i> .....	279

## Coordination and Negotiation

Coordination of Concurrent Scenarios in Multi-agent Interaction <i>Rie Tanaka, Hideyuki Nakanishi, Toru Ishida</i> .....	293
A Multi-agent Negotiation Model Applied in Multi-objective Optimization <i>Chuan Shi, Jiewen Luo, Fen Lin</i> .....	305
Model for Negotiating Prices and Due Dates with Suppliers in Make-to-Order Supply Chains <i>Lanshun Nie, Xiaofei Xu, Dechen Zhan</i> .....	315
Interest-Based Negotiation as an Extension of Monotonic Bargaining in 3APL <i>Philippe Pasquier, Frank Dignum, Iyad Rahwan, Liz Sonenberg</i> .....	327

## Agent Learning

Multiagent Reinforcement Learning for a Planetary Exploration Multirobot System <i>Zheng Zhang, Shu-gen Ma, Bing-gang Cao, Li-ping Zhang, Bin Li</i> .....	339
An Improved Multi-agent Approach for Solving Large Traveling Salesman Problem <i>Yu-An Tan, Xin-Hua Zhang, Li-Ning Xing, Xue-Lan Zhang, Shu-Wu Wang</i> .....	351
Design of Agent Registry/Repository System Based on ebXML <i>Il Kwang Kim, Jae Young Lee, Il Kon Kim</i> .....	362
Ant Agent-Based QoS Multicast Routing in Networks with Imprecise State Information <i>Xin Yan, Layuan Li</i> .....	374

## Peer to Peer Computing

Cactus: A New Constant-Degree and Fault Tolerate P2P Overlay <i>Chao Shui, Huaiming Wang, Pen Zhou, Yan Jia</i> .....	386
MPSS: A Multi-agents Based P2P-SIP Real Time Stream Sharing System <i>DeGuo Yang, Hui Wang, CuiRong Wang, Yuan Gao</i> .....	398
Dynamic Context Aware System for Ubiquitous Computing Environment <i>Seungkeun Lee, Junghyun Lee</i> .....	409

## Applications

Partial Group Session Key Agreement Scheme for Mobile Agents in e-Commerce Environment <i>Hyun-jin Cho, Gu Su Kim, Young Ik Eom</i> .....	420
Optimal Agendas for Sequential English Auctions with Private and Common Values <i>Yu-mei Chai, Zhong-feng Wang</i> .....	432

Intelligent Game Agent Based Physics Engine for Intelligent Non Player Characters <i>Jonghwa Choi, Dongkyoo Shin, Dongil Shin</i> . . . . .	444
---	-----

Palmprint Recognition Based on Improved 2DPCA <i>Junwei Tao, Wei Jiang, Zan Gao, Shuang Chen, Chao Wang</i> . . . . .	455
--	-----

## Short Papers

A Combination Framework for Semantic Based Query Across Multiple Ontologies <i>Yinglong Ma, Kehe Wu, Beihong Jin, Wei Li</i> . . . . .	463
--	-----

Adaptive Mechanisms of Organizational Structures in Multi-agent Systems <i>Zheng-guang Wang, Xiao-hui Liang, Qin-ping Zhao</i> . . . . .	471
--	-----

An Agent-Based Services Composition Framework for Ubiquitous Media <i>Hui Wang, Yuhui Zhao, Deguo Yang, Cuirong Wang, Yuan Gao</i> . . . . .	478
---	-----

A Multi-subset Possible World Semantics for Intention Operator of Agent <i>Shan-Li Hu, Chun-Yi Shi</i> . . . . .	484
--	-----

A Concurrent Agent Model Based on Twin-Subset Semantic <i>Youmin Ke, Shanli Hu</i> . . . . .	490
---	-----

The Communication Model of Migrating Workflow System <i>Zhaoxia Lu, Dongming Liu, Guangzhou Zeng, Gongping Yang</i> . . . . .	496
--	-----

Multi-user Human Tracking Agent for the Smart Home <i>Juyeon Lee, Jonghwa Choi, Dongkyoo Shin, Dongil Shin</i> . . . . .	502
---	-----

Towards Embedding Evolution into a Multi-agent Environment <i>Chantelle S. Ferreira, Elizabeth M. Ehlers</i> . . . . .	508
---	-----

A Multi-agent Framework for Collaborative Product Design <i>Jian Xun Wang, Ming Xi Tang</i> . . . . .	514
--	-----

Research on Algorithms of Gabor Wavelet Neural Network Based on Parallel Structure <i>Tingfa Xu, Zefeng Nie, Jianmin Yao, Guoqiang Ni</i> . . . . .	520
---	-----

A Momentum-Based Approach to Learning Nash Equilibria <i>Huaxiang Zhang, Peide Liu</i> . . . . .	528
---	-----



Model of Emotional Agent <i>Jun Hu, Chun Guan, Maoguang Wang, Fen Lin</i> .....	534
Multi Region-Tree Based Dynamic Commission Home Proxy Communication Mechanism for Mobile Agent <i>Zehua Zhang, Xuejie Zhang</i> .....	540
Research on Modeling and Description of Software Architecture of Cooperation-Oriented System <i>Munan Li, Hong Peng, Jinsong Hu</i> .....	546
An A-Team Based Architecture for Constraint Programming <i>Yujun Zheng, Lianlai Wang, Jinyun Xue</i> .....	552
The Efficient and Low Load Range Queries in P2P <i>Chao Shui, Pen Zhou, Yan Jia, Bing Zhou</i> .....	558
Research of Agent Based Multiple-Granularity Load Balancing Middleware for Service-Oriented Computing <i>Jun Wang, Di Zheng, Quan-Yuan Wu, Yan Jia</i> .....	564
Multiagent Model for Grid Computing <i>Qingkui Chen, Lichun Na</i> .....	571
Using Two Main Arguments in Agent Negotiation <i>Jinghua Wu, Guorui Jiang, Tiyun Huang</i> .....	578
A Frustum-Based Ocean Rendering Algorithm <i>Ho-Min Lee, Christian Anthony L. Go, Won-Hyung Lee</i> .....	584
A Model of Video Coding Based on Multi-agent <i>Yang Tao, Zhiming Liu, Yuxing Peng</i> .....	590
PDC-Agent Enabled Autonomic Computing: A Theory of Autonomous Service Composition <i>Bei-shui Liao, Li Jin, Ji Gao</i> .....	596
QoS Based Routing in Wireless Sensor Network with Particle Swarm Optimization <i>Xi-huang Zhang, Wen-bo Xu</i> .....	602
A Novel Multi-agent Automated Negotiation Model Based on Associated Intent <i>Weijin Jiang, Yusheng Xu</i> .....	608

Research on Design and Implementation of Adaptive Physics Game Agent for 3D Physics Game <i>Jonghwa Choi, Dongkyoo Shin, Dongil Shin</i> .....	614
An Improved TTS Model and Algorithm for Web Voice Browser <i>Rikun Liao, Yuefeng Ji, Hui Li</i> .....	620
Network-Based Face Recognition System Using Multiple Images <i>Jong-Min Kim, Hwan-Seok Yang, Woong-Ki Lee</i> .....	626
Reusable Component Oriented Agents: A New Architecture <i>W.H. Boshoff, E.M. Ehlers</i> .....	632
Expected Utility Maximization and Attractiveness Maximization <i>Ka-man Lam, Ho-fung Leung</i> .....	638
Modeling Negotiation in Combinatorial Auctions Based on Multi-agent <i>Man-Yin Shi, Shan-Li Hu</i> .....	644
A Methodology for Agent Oriented Web Service Engineering <i>Hongen Lu, Manish Chhabra</i> .....	650
Deliberate Soccer Agents Powered by Resource-Bounded Argumentation <i>Takumi Nisikata, Hajime Sawamura</i> .....	656
Agent-Oriented Probabilistic Logic Programming with Fuzzy Constraints <i>Jie Wang, Chunnian Liu</i> .....	664
An Agent-Based Adaptive Monitoring System <i>Sungju Kwon, Jaeyoung Choi</i> .....	672
A Peer-to-Peer CF-Recommendation for Ubiquitous Environment <i>Hyea Kyeong Kim, Kyoung Jun Lee, Jae Kyeong Kim</i> .....	678
Platform-Level Multiple Sensors Simulation Based on Multi-agent Interactions <i>Xiong Li, Kai Wang, Xianggang Liu, Jiuting Duo, Zhiming Dong</i> . . . .	684
Modeling Viral Agents and Their Dynamics with Persistent Turing Machines and Cellular Automata <i>Jingbo Hao, Jianping Yin, Boyun Zhang</i> .....	690
A Lightweight Architecture to Support Context-Aware Ubiquitous Agent System <i>Qiu-sheng He, Shi-liang Tu</i> .....	696

Towards an Agent-Based Robust Collaborative Virtual Environment for E-Learning in the Service Grid <i>Changqin Huang, Fuyin Xu, Xianghua Xu, Xiaolin Zheng . . . . .</i>	702
Design of Music Recommendation System Using Context Information <i>Jong-Hun Kim, Chang-Woo Song, Kee-Wook Lim, Jung-Hyun Lee . . .</i>	708
Semantic Grid: Interoperability Between OWL and FIPA SL <i>Maruf Pasha, H. Farooq Ahmad, Arshad Ali, Hiroki Suguri . . . . .</i>	714
An Agent-Based Adaptive Task-Scheduling Model for Peer-to-Peer Computational Grids <i>Zhikun Zhao, Wei Li . . . . .</i>	721
Cluster-Based Secure Data Transmission Solution for Ad Hoc Network <i>Hwan-Seok Yang, Joung-Min Kim, Seung-Kyu Park . . . . .</i>	728
Embodied Conversational Agent Based on Semantic Web <i>Mikako Kimura, Yasuhiko Kitamura . . . . .</i>	734
Dynamic Service Composition Model for Ubiquitous Service Environments <i>Seungkeun Lee, Junghyun Lee . . . . .</i>	742
Framework for Agent-Based Buying Decision Process <i>Sazalinsyah Razali, Mashanum Osman . . . . .</i>	748
Improving Adaptability and Transparency of Dynamically Changing Mobile Agent Runtime Environments <i>JinHo Ahn, SungMin Hur . . . . .</i>	754
AgentAssembly: The Agent Framework Platform <i>Ockmer L. Oosthuizen, E.M. Ehlers . . . . .</i>	760
A Multi-agent Architecture for CSCW Systems: From Organizational Semiotics Perspective <i>Wenge Rong, Kecheng Liu . . . . .</i>	766
Knowledge Description Model for MAS Utilizing Distributed Ontology Repositories <i>Kyengwhan Jee, Jung-Jin Yang . . . . .</i>	773
Object Recognition Using K-Nearest Neighbor in Object Space <i>Jong-Min Kim, Jin-Kyoung Heo, Hwan-Seok Yang, Mang-Kyu Song, Seung-Kyu Park, Woong-Ki Lee . . . . .</i>	781

Research on Smart Multi-agent Middleware for RFID-Based Ubiquitous Computing Environment <i>Minwoo Son, Joonhyung Kim, Dongil Shin, Dongkyoo Shin</i> . . . . .	787
Certificate Management System in MANET for Ubiquitous Computing <i>Dae-Young Lee, Sang-Hyun Bae</i> . . . . .	793
FPGA Based Intrusion Detection System Against Unknown and Known Attacks <i>Dong-Ho Kang, Byoung-Koo Kim, Jin-Tae Oh, Taek-Yong Nam, Jong-Soo Jang</i> . . . . .	801
Agent-Based Real Time Intrusion Detection System Against Malformed Packet Attacks <i>Jun-Cheol Jeon, Eun-Yeung Choi, Kee-Young Yoo</i> . . . . .	807
Efficient Mutual Authentication Scheme with Smart Card <i>Eun-Jun Yoon, Kee-Young Yoo</i> . . . . .	813
Strong Mobility for FIPA Compliant Multi-agent Systems <i>Javed Iqbal, H. Farooq Ahmad, Arshad Ali, Hiroki Suguri, Sarmad Sadik</i> . . . . .	819
<b>Author Index</b> . . . . .	825

# Agent and Grid Technologies for Intercultural Collaboration

Toru Ishida

Department of Social Informatics, Kyoto University  
ishida@i.kyoto-u.ac.jp

## 1 Introduction

After September 11, we remember there was a clear conflict in public opinions within western countries. While 77% of those interviewed in France opposed to military intervention against Iraq (2003.1.9 Le Figaro), 63% of the U.S. population were proud of the U.S. role in the war (2003.3.22 CBS News). Conflicts in governmental policies are common, but conflicts in public opinion between western countries have not been observed before. Though we all share information on the Web, similar conflicts arose recently in East Asia. While about 90 percent of Chinese polled blamed Japan, more than half of Japanese polled said it was hard to tell who bore responsibility (2005.8.24 Genron NPO and Peking University). According to Global Reach, the ratio of English speaking people online has decreased to 35.2% in 2004. To increase mutual understanding between different cultures and of opinions in different languages, it is essential to build a language infrastructure on top of the Internet.

Motivated by the above goal, we conducted Intercultural Collaboration Experiments in 2002 (ICE2002) with Chinese, Korean and Malaysian colleagues [4]. We thought that machine translation would be useful in facilitating intercultural experiments. We gathered machine translators to cover five languages: Chinese, Japanese, Korean, Malay, and English. More than forty students and faculty members from five universities in four countries joined this experiment. The goal was to develop open source software using the participants' first language. The experiment started in April 2002 and ended in December 2002, and a total of 31,000 messages were collected.

## 2 Translation Agent

The first lesson learned from ICE2002 is the impact of machine translation quality. Since it was not easy to understand translated sentences as they appeared on multilingual BBS, users repeatedly tried to refine the translation results before posting the message. We call this behavior *self-initiated repair*. Each time the input sentence was modified, its content and meaning changed. Nuance or emotions within the sentences were lost, and a joke became a formal description. A comparative analysis was performed on the efficiency of self-initiated repair: when repairs were done with input Japanese sentences, the quality of output English sentences was largely improved. However, output sentences in Chinese and Korean were not improved.

This is because users modified the input Japanese sentences based on output English sentences, since the languages of neighboring countries are not taught in Asia.

Machine translation systems developed for written texts do not seem good at translating spoken languages. As in Figure 1, machine translation research has taken a transparent channel metaphor so far: one end of the channel says “Hello,” and the other end hears “Moshi Moshi.” Naturally, the noise ratio became an evaluation measure of the channel, and the improvement of translation quality has been the dominant research goal. Since the level of error is not acceptable, however, users are forced to rephrase the input text messages. Furthermore, users hardly know how to rephrase the text in order to get a good translation.

To solve such grounding problems, why doesn’t the translation system simply say, “I can’t translate it”? What if we replace *accuracy* as the goal of machine translators with their *interactivity*? Interactivity includes the ability to state “I don’t understand,” or “please rephrase this sentence.” The key to increase interactivity lies in the meta-level architecture: “To know that we know what we know, and to know that we do not know what we do not know, that is true knowledge.” This ability will allow interaction between users and machine translators, which we call *translation agents*, to improve grounding and permit the negotiation of meaning.

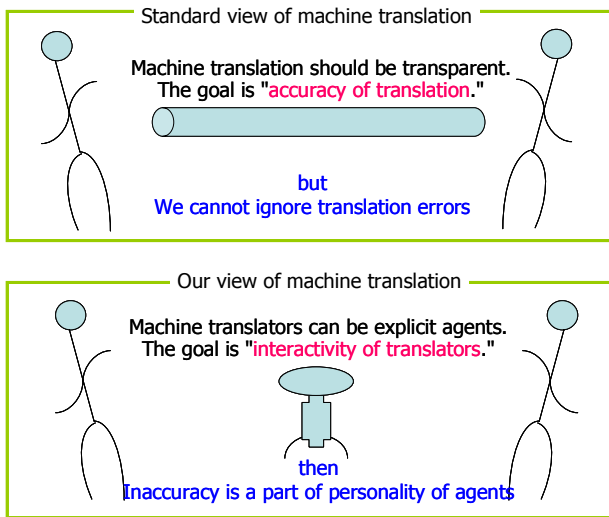


Fig. 1. Language Grid Architecture

The first step towards translation agents with a meta-level architecture is to make machine translators to understand their own quality. To achieve this, we have developed back translation with a similarity measure between input and output sentences [5]. If we use Japanese-Chinese-Japanese translation, users can repair the input Japanese sentence based on the output Japanese sentence to improve the quality of the intermediate Chinese sentence. If the input and output sentences are highly similar, the machine translators can validate their work by themselves. If not, the

translator simplifies the input sentence until it obtains a high quality result. The translator then determines which word or phrase is hard to translate [6]. By viewing a translation agent as a meta-level architecture, the agent can be extended to prevent misconceptions caused by low translation quality [7,8], or we can develop *intercultural conversational agents* that provide not only translations but also cultural interpretations of original messages [3].

### 3 Language Grid

We provided translation services covering five languages for ICE2002. From this experiment, we found that language services are often not *accessible*, because of intellectual property rights and cost. We tend to think that effective language infrastructures have been already developed, since we can use machine translations to view Web pages. However, if one tries to create new services by combining existing language services, he/she is soon forced to face the realities: the language services available come with different contracts and prices. Contracts can be complex because of the concern over intellectual property rights. Prices can be high, and no explanation is available. Furthermore, language services are often not *usable*, because of unstandardized interfaces. Users have to develop different wrappers for different language services. There is no quality assurance for machine translators. Users have to estimate their quality by themselves. Existing services are often not customizable: machine translators seldom allow users to modify them; it is hard to add new words to users' dictionaries.

To increase the accessibility and usability of language services, we proposed the *language grid*, which treats existing language services as atomic components and enables users to create new language services by combining the appropriate components [2].

The language grid has two different goals. One is to connect existing online language services that cover nations' standard languages. Those services are created often by linguistic professionals with the support of their governments. Typical examples include online dictionaries and translation services. Another goal is to assist users to create new language services, which are often related to intercultural activities in their local community. Consequently, the language grid consists of two different types of service networks as shown in Figure 2.

The *horizontal language grid* combines existing language services using semantic Web technology. For example, by connecting WordNet and an English-Japanese dictionary, we can create WordNet with a Japanese interface. The horizontal language grid benefits a wide range of users by providing standard language services. The *vertical language grid*, on the other hand, layers community language services on the horizontal language grid to support intercultural activities. The *language service ontology* is introduced to represent entries of language resources and processing functions. The ontology also enables users to easily extend default service interfaces to define community-oriented services [1]. Suppose a nonprofit organization has its own parallel texts to support foreigners in a specific affiliated hospital. The organization can combine standard parallel texts created by medical doctors and its own resources by using the language grid.

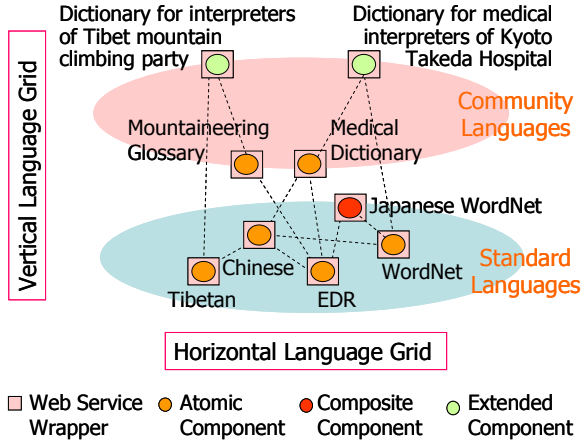


Fig. 2. Language Grid Architecture

## 4 Conclusion

To support intercultural collaboration, this paper proposes *translation agents* with a meta-level architecture to increase the *interactivity* of language services, and the *language grid* to increase their *accessibility* and *usability*. As is clear, field study is essential in researching the language grid and understanding how language services are to be created in local communities.

## References

- [1] Y. Hayashi and T. Ishida. A Dictionary Model for Unifying Machine Readable Dictionaries and Computational Concept Lexicons. *LREC*, 2006.
- [2] T. Ishida. Language Grid: An Infrastructure for Intercultural Collaboration. *SAINT*, pp. 96-100, 2006.
- [3] T. Ishida. Ubiquitous Cultural World. Future of AI. *IEEE Intelligent Systems*, 2006.
- [4] S. Nomura et al. Open Source Software Development with Your Mother Language: Intercultural Collaboration Experiment 2002. *HCI*, pp. 1163-1167, 2003.
- [5] K. Ogura et al. User Adaptation in MT-mediated Communication. *IJCNLP*, pp.596-601, 2004.
- [6] K. Uchimoto et al. Automatic Rating of Machine Translatability, *MT Summit X*, pp. 235-242, 2005.
- [7] N. Yamashita and T. Ishida. Automatic Prediction of Misconceptions in Multilingual Computer-Mediated Communication. *IUI*, pp.62 - 69, 2006.
- [8] N. Yamashita and T. Ishida. Effects of Machine Translation on Collaborative Work, *CSCW*, 2006.



# Agent Grid Collaborative Environment

Zhongzhi Shi

Key Laboratory of Institute of Intelligent Information Processing,  
Institute of Computing Technology  
Chinese Academy of Sciences, Beijing 100080, China  
shizz@ics.ict.ac.cn

**Abstract.** Collaborative environment will provide the architecture and infrastructure that allows seamlessly integrated comprehensive support services for collaboration involving sharing data, information, knowledge, tools for manipulation, representation and visualizations. Agent Grid Intelligence Platform (AGrIP) will aid people, teams and group to collaborate with searching processes, mining processes, decision processes and action processes. AGrIP is a highly open software environment whose structure is capable of dynamical changes. It's a loosely coupled computer network of ever expanding size and complexity. It can be viewed as a large, distributed information resource, with nodes on the network designed and implemented by different organizations and individuals with widely varying agendas. The four-layer model for AGrIP is presented, that is data resources, multi-agent, middleware and applications.

In this presentation I will emphasize to discuss agent model, AGrIP architecture and collaborative strategies. For collaborative strategies we have proposed flexible working flow, role assignment, ontology-based collaboration, policy driven and planning.

# An Agent-Based System Integration Architecture for Intelligent Service Robots

Jaeho Lee

Dept. of Electrical and Computer Engineering,  
The University of Seoul  
90 Cheonnong-dong, Tongdaemun-gu, Seoul 130-743, Korea  
jaeho@uos.ac.kr

**Abstract.** While industrial robots typically perform pre-programmed and routine manufacturing jobs for humans, intelligent service robots serve and interact with people intelligently with its own perception and cognition of external circumstance. As an intelligent system, intelligent service robots are also characterized by the ability to evolve and learn to increase their performance over time.

Recently intelligent service robots have received great attention as one of the next-generation growth engine in many countries including Korea. In this presentation, I introduce the research activities on intelligent service robots in Korea and then provide an agent-based approach to the integration of various distributed functionalities to overcome the inherent complexity of intelligent service robots in dynamic environments.

# DartGrid: A Semantic Grid and Application for Traditional Chinese Medicine

Zhaohui Wu

College of Computer Science, Zhejiang University, Hangzhou 310027, China  
wzh@zju.edu.cn

## 1 Introduction

The rapid growth of web along with the increasing decentralization of organizational structures has led to the creation of a vast interconnected network of distributed electronic information in many fields such as medical science, bioinformatics, high-energy physics etc. The data produced and the knowledge derived from it will lose value in the future if the mechanisms for sharing, integration, cataloging, searching, viewing, and retrieving are not quickly improved. Building upon techniques from both Semantic Web [1] and Grid [2] research areas, we propose the DartGrid [3][4] which exhibits a Dynamic, Adaptive, RDF-mediated and Transparent (DART) approach for building semantic grid applications.

## 2 Abstract Architecture

The DartGrid is designed and developed as a two-layer service architecture: basic service layer and semantic service layer (as Fig.1).

What we mainly contribute is at the semantic service level. The services at this level are mainly designed for semantic-based relational schema mediation and semantic query processing.

1. Ontology Service [5] exposes the shared ontologies that are defined using RDF/OWL languages and the ontologies are used to mediate heterogeneous relational databases.
2. Semantic Registration Service [6] establishes the mappings from source relational schema to mediated ontologies.
3. Semantic Query Service [7] is used to process SPARQL semantic queries over heterogeneous databases.

## 3 Application in TCM

Currently, the DartGrid has been deployed at China Academy of Traditional Chinese Medicine (CATCM) [8] and currently provides access to over 70 databases including TCM herbal medicine databases, TCM compound formula databases, clinical

symptom databases, traditional Chinese drug database, traditional Tibetan drug database, TCM product and enterprise databases, and so on. The current ontology used in the DartGrid is the world’s largest TCM ontology [9], which includes more than 20,000 concepts and 100,000 individuals and the ontology under development is still part of the complete project. Besides the DartGrid provides many high-level solutions like semantic search, semantic portal, knowledge discovery in databases (KDD), problem solving of multi-agents [10], etc., to fulfil different requirements of scientific research in TCM. In general, users from CATCM reacted positively to our system.

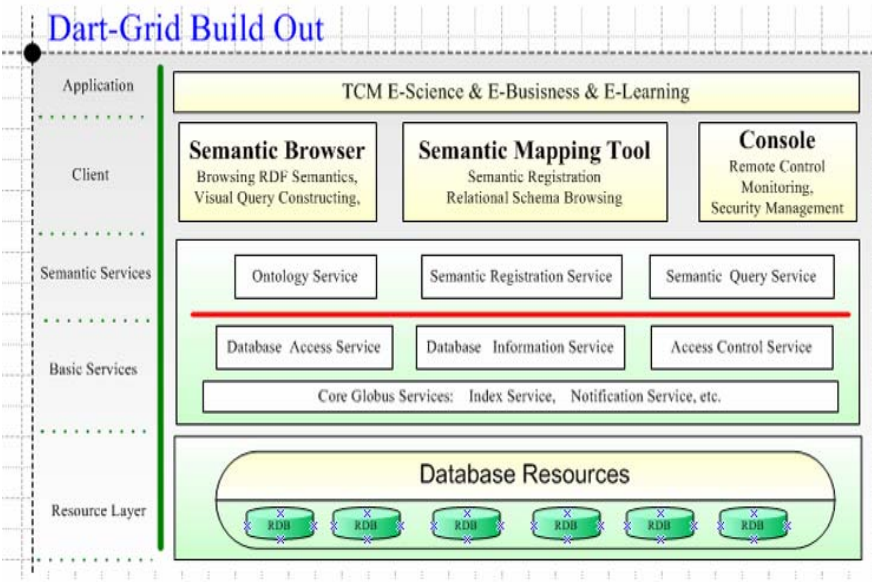


Fig. 1. The layered structure of the DartGrid

## 4 Future Work

Although the DartGrid has been used successfully in TCM, practical scalability still needs to be tested if the number of databases becomes larger. We will continue to build the TCM ontology and enrich the semantics of the DartGrid to support more applications in TCM. We also plan to extend and apply the DartGrid to more fields like Intelligent Transportation Systems (ITS) [11], e-Business and so on.

## References

1. Berners-Lee T., Hendler J., Lassila, O., The semantic web, Scientific American, Vol. 279, No. 5, pp. 34–43, 2001.
2. Foster I., Kesselman C., Tuecke S., The Anatomy of the Grid: enabling scalable virtual organizations, Lecture Notes in Computer Science, Vol. 2150, pp.1–26, 2001.

3. Zhaohui Wu, Huajun Chen, Chang Huang et al, DartGrid: Semantic-based Database Grid, 4th International Conference on Computational Science (ICCS 2004), Kraków, Poland, Jun. 2004.
4. Huajun Chen, Zhaohui Wu, Yuxing Mao, Guozhou Zheng, DartGrid: a Semantic Infrastructure for Building Database Grid Applications, *Concurrency Computat: Pract. Exper*, 2000.
5. Zhaohui Wu, Yuxin Mao, Huajun Chen, Zhao Xu, An Ontology and Context based Client Model for Dart Information Grid, *Int. J. High Performance Computing and Networking*, vol. 3, No. 4, 2005
6. Huajun Chen, Zhaohui Wu, Guozhou Zheng, Yuxing Mao, RDF-Based Schema Mediation for Database Grid, 5th IEEE/ACM International Workshop on Grid Computing (Grid Computing 2004), Pittsburgh, USA, Nov. 2004.
7. Huajun Chen, Zhaohui Wu, Yuxing Mao, Q3: A Semantic Query Language For Dart Database Grid, *Lecture Notes in Computer Science* 3251: 372-380, 2004.
8. Huajun Chen, RDF-based Relational Database Integration and its Application in Traditional Chinese Medicine, The 22nd International Conference on Data Engineering, Atlanta, GA, 2006.
9. Xuezhong, Zhou, Zhaohui Wu et al, Ontology Development for Unified Traditional Chinese Medical Language System, *Journal of Artificial Intelligence in Medicine*, vol.32, issue 1, pages 15-27, Sep. 2004.
10. Yuxin Mao, WK Cheung, Zhaohui Wu, et al. Dynamic Sub-Ontology Evolution for Collaborative Problem-Solving. *AAAI Fall Symposium*, v FS-05-01, p 1-8, 2005.
11. Zhaohui Wu et al, DartGrid II: A Semantic Grid Platform for ITS, *IEEE Intelligent Systems*, vol. 20, Issue 3, pp. 12-15, Jun. 2005.

# A Grammatical Framework for Modelling Multi-agent Dialogues

Gemma Bel-Enguix, María Adela Grando\*, and M. Dolores Jiménez-López

Research Group on Mathematical Linguistics,  
Rovira i Virgili University  
Pl. Imperial Tàrraco 1, 43005 Tarragona, Spain  
Tel.: +34 977559543; Fax: +34 977 559597  
{gemma.bel, mariadolores.jimenez}@urv.net,  
mariaadela.grando@estudiants.urv.es

**Abstract.** In this paper we present a variant of grammar system from Formal Language theory, that seems to be appropriate for modelling dialogues in multi-agent systems. Previous attempts of simulating conversations in grammar systems used rewriting rules, which remained fix and unchangeable during the whole dialogue. The novelty of this approach is that taking inspiration from a Multi Agent Protocol language we define a grammatical system in which agents(grammars) behaviours are given by string process descriptions or protocols. Strings can be modified during running time, allowing to dynamically alter agent behaviour according to the environmental changes. This provides agents with a more flexible and adaptative reaction to unpredictable and changeable conversational space.

**Keywords:** Multi-agent Systems, Interaction Protocols, Grammar Systems, Reproductive Eco-Grammar Systems.

## 1 Introduction

The theory of *Grammar Systems* was developed as a grammatical model for distributed computation. Briefly a grammar system is a finite set of grammars working together according to a specified protocol of cooperation, to generate one language. Many variants of grammar systems have been developed and studied as language generators, simulators of natural or artificial environments, problem solvers and conversational models. With respect to the last approach some theoretical variants of grammar systems were introduced in [4,7,8,2], that proved to be appropriate for simulating human dialogue. But the mentioned models were presented as purely theoretical models, without implementation details or examples of empirical applications. The purpose of this paper is to introduce a theoretical model that allows to simulate conversational environments more generic than human dialogues. Here we introduce an extension of the so-called *Reproductive Eco-grammar system* [5] that seems to allow to model multi agent

---

\* This work was possible thanks to the research grant “Programa Nacional para la Formación del Profesorado Universitario”, from the Ministry of Education, Culture and Sports of Spain.

dialogues. The model presented in this paper inherits all the features that according to [8] make grammar systems appropriate to model conversation: distribution, modularity, parallelism, interaction, coordination and emergent behaviour. The novelty of this approach is given by the way agent participation in the dialogue is represented, described and modified during the conversational process. In the grammatical traditional approach agent behaviour was given by rewriting rules that were used to depict how the agent interact with other agents and modify the conversational space. In the framework introduced here agent participation in the dialogue is given by a process description of the agent behaviour, called *protocol*, which is stored in a string. The idea of using protocols for describing cooperation and interchange of messages between agents was taken from the *Multi Agent Protocol (MAP) language* introduced in [10] and with practical applications in Multi-agent field. In particular the model presented here uses  $MAP^a$  [6], an extension of *MAP*, because it has a number of additional features which provide more flexibility and dynamism than the original language. According to  $MAP^a$ , protocols define social norms that agents must observe if they are to participate in a conversation. The use of protocols ensures that the interactions between agents in a multi-agent system happen in a controlled and predictable manner. Nonetheless, these protocols do not overly restrict the individual autonomy of the agents. Agents can still choose next actions to perform (subprocesses to execute) by decision procedures, showing clever behaviour.  $MAP^a$  language allows also to define protocols where agents do not know in advance how to behave in particular situations or how to perform certain tasks. In this case agents can ask other agents for advice and once they have received a message containing the corresponding protocol, they can perform it. By defining interaction protocols during run-time, agents are able to interact in systems where it is impossible or impractical to the define the protocol beforehand. These features from  $MAP^a$  language combined with the fact that strings are used as a physical medium to store agent protocol provide agents (grammars) with dynamic and flexible reaction to environmental changes and unexpected situations. The model presented in this paper takes advantage of the nature of strings that can be modified during the conversational process by rewriting rules, and uses strings to save agent current mental state and protocol. Therefore the use of rewriting rules is restricted to the action of modifying during conversation the strings that describe the state of participation of the agent in the dialogue and the agent performing protocol.

The paper is organized in the following way. In section 2, we introduce a definition of an extension of Reproductive Eco-Grammar Systems, the so-called Extended Reproductive Eco-Grammar Systems (EREG). In section 3 we present the syntax of the  $MAP^a$  language and an example of its used in the definition of a particular multi agent conversational scene, a sale. In section 4 we provide an implementable definition of EREG system that turns out to be able to model any conversational scenario that can be modelled by  $MAP^a$  protocols in multi-agent systems. Finally, in section 5 and 6 we discuss improvements introduced by our model with respect to other approaches and we present proposals of future work.

## 2 Extended Reproductive Eco-Grammar System

In this section, we perform some modifications to the definition of Reproductive Eco-grammar (REG) systems introduced in [5].

The first modification is related to the actions that an agent performs to modify the state of the system. In REG systems agents could only rewrite one symbol from the environmental state using one *context free* rule  $V \rightarrow w$ , where  $V$  is a symbol over some alphabet and  $w$  is a string of symbols over the same alphabet. While in the extended version proposed here agents are allowed to perform as many action rules as they select to change the state of the environment and their own state. Mainly an agent  $A_i$  can select from  $R_i$  by mapping  $\psi_i$  a set of action rules  $R_1$  to change its current state  $\omega_i$  and also a set of action rules  $R_2$  for rewriting symbols from the environmental state  $\omega_E$ . The purpose of this change is to model the fact that the decision of an agent to participate in the conversation (change the state of the environment) is triggered by a change in his mental state.

The second modification regards the possible conflict that action rules selected by mapping  $\psi_i$  can introduce with respect to environmental rules  $P_E$ . A potential conflict can also exist when  $\psi_i$  and  $\varphi_i$  modify  $\omega_i$ . The criterion to avoid this difficulty is the following: in both cases action rules selected by mapping  $\psi_i$  are applied first, and then rules from  $P_E$  and rules selected by mapping  $\varphi_i$  are applied respectively in  $\omega_E$  and  $\omega_i$  over those symbols that were not rewritten in the first step.

Taking into account the above modifications, we formally define Extended Reproductive Eco-Grammar Systems as follows.

**Definition 1.** *An Extended Reproductive Eco-Grammar (EREG) system is an  $(n+1)$ -tuple  $\Sigma = (E, \mathcal{A}_1, \dots, \mathcal{A}_n)$  where:*

- $E = (V_E, P_E)$  is the environment with  $V_E$  a finite alphabet and  $P_E$  a finite set of 0L rewriting rules over  $V_E$ ;
- $\mathcal{A}_i$  is a multiset (a set whose elements can occur several times each), called the population of the agents of the  $i$ -th type,  $1 \leq i \leq n$ , where every agent  $A_i$  in  $\mathcal{A}_i$  has the same form  $A_i = (V_i \cup \{\sqcup\}, P_i, R_i, \varphi_i, \psi_i)$ , where:
  - $V_i$  is a finite alphabet,  $\sqcup$  is the reproduction symbol that can occur only on the right-hand side of productions of  $P_i$ ,  $P_i$  is a finite set of 0L rewriting rules over  $V_i$ , and  $\varphi_i : V_E^* \rightarrow 2^{P_i}$  has the same interpretation as in REG systems.
  - $R_i$  is a finite set of 0L rewriting rules  $x \rightarrow y$  over the environment ( $x \in V_E$  and  $y \in V_E^*$ ) or over the state of the agent ( $x \in V_i$  and  $y \in V_i^*$ ).
  - $\psi_i : V_i^+ \rightarrow 2^{R_i}$  has the same interpretation as in REG systems.

**Definition 2.** *(System configuration) A configuration of an EREG system  $\Sigma = (E, \mathcal{A}_1, \dots, \mathcal{A}_n)$  is an  $(n+1)$ -tuple:  $\sigma = (\omega_E, W_1, W_2, \dots, W_n)$  where:  $\omega_E \in V_E^*$  and  $W_i$  is a multiset of strings  $\omega_{i1}, \dots, \omega_{ik_i}$ , where  $\omega_{ij} \in V_i^+$  for  $1 \leq j \leq k_i$ ,  $1 \leq i \leq n$ ;  $\omega_E$  is the current evolution state of the environment and the strings  $\omega_{i1}, \dots, \omega_{ik_i}$  correspond to the evolution states of all currently existing agents of the  $i$ -th type.*



**Definition 3.** (*Agent Derivation*) Agent state  $\omega_i$  derives to new state  $\omega'_i$  denoted by  $\omega_i \vdash \omega'_i$  iff  $\omega_i \vdash_1 \beta'$  as the result of parallel application of all the environment rewriting rules selected by mappings  $\psi_i$  from agent  $A_i$  and  $\beta' \vdash_2 \omega'_i$  according to mapping  $\varphi_i$ .

**Definition 4.** (*Environment Derivation*) Environmental state  $\omega_E$  derives to new state  $\omega'_E$  denoted by  $\omega_E \models \omega'_E$  iff  $\omega_E \models_1 \alpha'$  as the result of parallel application of all the environment rewriting rules selected by mappings  $\psi_i$  from all agents and  $\alpha' \models_2 \omega'_E$  according to  $P_E$ .

**Definition 5.** (*System derivation*) Let  $\Sigma = (E, \mathcal{A}_1, \dots, \mathcal{A}_n)$  be an EREG system and let  $\sigma = (\omega_E, W_1, W_2, \dots, W_n)$  and  $\sigma' = (\omega'_E, W'_1, W'_2, \dots, W'_n)$  be two configurations of  $\Sigma$ . We say that  $\sigma$  is directly changed for (it directly derives)  $\sigma'$ , denoted by  $\sigma \Longrightarrow_{\Sigma} \sigma'$ , iff  $\omega'_E$  arises from  $\omega_E$  by evolution of the environment affected by all agents being active in  $\sigma$ , and  $W'_i$  is the reproduction of the states actually evolving from the states in  $W_i$ ,  $1 \leq i \leq n$ . Note that action rules have priority over evolution rules.

The complete definition of Reproductive Eco-Grammar Systems can be found in [5]. For unexplained notions related to formal language theory the reader is referred to [9].

Up to now we have presented the definition of EREG systems. In the next section, we introduce the formal syntax of the extended Multi Agent Protocol  $MAP^a$ .

### 3 $MAP^a$ Syntax

$MAP^a$  [6] is a formalism for the expression of protocols, which allows to model social interactions between groups of agents. The language is a sugared process-calculus. The key concepts in  $MAP^a$  are *scenes* and *roles*. A *scene* can be thought

$S$	$::= \langle R^{(i)}, P^{(i)}, M^{(k)}, K^{(b)} \rangle$	(Scene)
$R$	$::= \langle id, Proc^{(l)}, K^{(m)}, r^{(n)} \rangle$	(Role)
$P$	$::= \mathbf{agent}(id, r, Proc^{(l)}, K^{(m)}, \phi^{(f)}) = op.$	
$K$	$::= \mathbf{axiom}$	(Knowledge)
$Proc$	$::= \mathbf{type} :: id((\phi, type)^{(g)})$	(Procedure)
$M$	$::= id((\phi, type)^{(h)})$	(Performative)
$op$	$::= v$	(Variable)
	$\alpha$	(Action)
	$op_1 \mathbf{then} op_2$	(Sequence)
	$op_1 \mathbf{or} op_2$	(Choice)
	$(op)$	(Precedence)
$\alpha$	$::= \mathbf{null}$	(No Action)
	$v = p(\phi^{(g)})$	(Decision)
	$id(\phi^{(x)}) \leftarrow \mathbf{agent}(id, r)$	(Receive)
	$id(\phi^{(y)}) \Rightarrow \mathbf{agent}(id, r)$	(Send)
	$\mathbf{agent}(id, r, Proc^{(w)}, K^{(v)}, \phi^{(d)})$	(Invocation)
$\phi$	$::= c \mid - \mid v$	(Term)

**Fig. 1.**  $MAP^a$  Language Syntax

of as a bounded space in which a group of agents interact on a single task. We assume that a scene places barrier conditions on the agents, such that a scene cannot begin until all the agents are present, and the agents cannot leave the scene until the dialogue is complete.

The concept of an agent *role* is also central to the definition of our protocols. Agents entering a scene assume a role which they can change during the scene. For example, a sale dialogue or scene may involve agents with the roles of *customer* and *clerk*. Roles are defined as a hierarchy, where more specialized roles appear further down in the graph of roles. For example, an agent may initially assume the role *clerk* but may change to the more specialized role *Section 1 clerk* during a scene. The role also identifies capabilities which the agent must provide. It is important to note that a protocol only contains operations which are specific to the mechanisms of communication and coordination between agents. This makes it straightforward to understand the operation of the protocol without extraneous details.

A BNF-style syntax for  $MAP^a$  is shown in Figure 1. Superscripts are used to indicate a set, e.g.  $P^{(i)}$  is a set with elements  $P$  of size  $i$ . A protocol in  $MAP^a$  is represented by a scene  $S$ . A scene comprises a *role hierarchy*  $R^{(i)}$ , a set of agent protocols  $P^{(i)}$  which are parameterized on these roles, a set of performatives  $M^{(k)}$  which define the *dialogic structure* and a set of axioms  $K^{(b)}$  which is the *common knowledge* in the scene. The role hierarchy is defined as a set of role definitions. Each of these definitions  $R$  has a unique identifier  $id$ , a set of decision procedures  $Proc^{(l)}$  which are shared within the role, a set of axioms  $K^{(m)}$  which are common to the role, and lastly a set of upper roles  $r^{(n)}$ , which appear above the role in the hierarchy. This set will be empty for a top-level role. A protocol  $P$  is defined by a unique identifier for the protocol  $id$ , a role  $r$ , a set of procedures  $Proc^{(l)}$  associated to the role, and a set of role axioms  $K^{(m)}$  and possibly other parameters  $\phi^{(f)}$ . Knowledge  $K$  is represented by axioms, facts which are believed to be true. The reasoning over this knowledge is performed by decision procedures  $Proc$  which are external to the protocol. In effect, the decision procedures provide an interface between the communicative and the rational process of the agent. We assume that the decision procedure has full access to the scene, role and private agent knowledge. The decision procedures take a set of  $\phi^{(g)}$  arguments and return a result which is bounded to a variable  $v$ . The core of the protocols are constructed from operations  $op$  that can be protocol variables  $v$ , operations to control the flow of the protocol, or actions  $\alpha$  which have side-effects and can fail. The main operations are sequence *then* or undeterministic choice *or*. The actions allow us to send and receive messages, to invoke external decision procedures and to create agent instances. We also include a null action. A message is defined as a performative identifier  $id$  together with a set of arguments. The performatives must be defined in the dialogic structure of the scene  $M^{(k)}$ . The origin or destination of a message is specified by an identifier  $id$  together with a role  $r$ . The invocation of an agent instance allows an agent to perform recursive calls (agent invocations with same

identifier and role), change of role (same identifier, different role) or create other agent instance (different identifier).

To make concepts clearer we introduce an example of  $MAP^a$  definition that we explain below.

### 3.1 An Example of $MAP^a$ Definition

In Figure 2 we present a  $MAP^a$  definition for a typical sale conversation with four role definitions: *Customer*, *Clerk*, *InformationSeeker (IS)* and *Information-Provider (IP)*. Customer role initiates the dialogue choosing a clerk to ask for an item  $t$  to buy and changing its role to provider. While the clerk role waits for a customer to ask for an item  $t$  to change to the seeker role.

The core of the dialogue is between a provider agent that wants to buy an item  $t$  and answers the questions asked by a seeker agent who inquires item characteristics to determine the product to sell. The results of the interchanges of questions  $q$  from the seeker agent and answers  $a$  from the provider agent are collected during the dialogue by both roles in sets  $Q$  of elements ( $q = a$ ).

The IP role is defined in five cases separated by the choice operator. The evaluation model of the protocol is such that the protocol will backtrack and retry each of these cases until one of them can be satisfied. In the first case, the information provider receives a request for answering a question  $q$  from an information seeker. The information provider will then update its state, incorporating ( $q = ?$ ) to  $Q$ , and attempt to answer the request. The appropriate answer is determined in cases 3, 4 and 5. Second case allows the agent to determine how long he is able to wait for a question from a seeker agent. Third case is triggered when the information provider has a direct answer  $a$  for the request  $q$ , in which case this answer is returned to the seeker, incorporating ( $q = a$ ) to  $Q$ . In the fourth case more information is required, so the provider becomes a seeker for this new information. Upon obtaining this extra information, the agent reverts back from a seeker to a provider again. In the final case, the provider receives from the seeker a protocol containing the appropriate actions to perform to buy the item  $t$  satisfying all the agreed characteristics from  $Q$ , and performs *prot*.

The information seeker (IS) protocol is essentially the complement of the information provider protocol, and there are five corresponding cases. In the first case, the seeker sends a request *newq* for information to the provider and updates its state incorporating (*newq = ?*) to  $Q$ . The second case deals with the response from the provider which actualises the set  $Q$  and restarts the protocol. Third case allows seeker to fix a deadline to wait for responses from the provider. In the fourth case agent decides to assume the role of provider to be more explicit and clearer about what he is asking for to the other agent. After providing the required information he reassumes its role of seeker. Lastly, he can decide that the collected information contained in  $Q$  is enough to determine the item to sell. In this case he informs the provider with the protocol variable *prot* the corresponding actions to buy it. The possibility of including a protocol variable provides the dialogue with flexibility and adaptability. For example in this case it is not necessary to consider beforehand all the possible buying protocols according to

<pre> <b>agent</b>(id<sub>1</sub>, <b>Customer</b>, <b>Proc</b><sup>(a)</sup>, <b>K</b><sup>(b)</sup>, <b>t</b>) id<sub>2</sub>=choose_clerk(t) then ask(t) ⇒ <b>agent</b>(id<sub>2</sub>,IS) then <b>agent</b>(id<sub>1</sub>, IP, <b>Proc</b><sup>(a)</sup>, <b>K</b><sup>(b)</sup> ∪ {Q=[]}) . </pre>	<pre> <b>agent</b>(id<sub>1</sub>,<b>Clerk</b>, <b>Proc</b><sup>(c)</sup>, <b>K</b><sup>(d)</sup>,<b>S</b>) ask(t) ⇐ <b>agent</b>(id<sub>1</sub>,<b>Customer</b>) then <b>agent</b>(id<sub>1</sub>, IS, <b>Proc</b><sup>(c)</sup>, <b>K</b><sup>(d)</sup> ∪ {Q=[]}) . </pre>
<pre> <b>agent</b>(id<sub>1</sub>, <b>IP</b>, <b>Proc</b><sup>(t)</sup>, <b>K</b><sup>(z)</sup> ∪ {Q=Q<sub>0</sub>}) [ request(q) ⇐ <b>agent</b>(id<sub>2</sub>, IS) then <b>agent</b>(id<sub>1</sub>, IP, ∅, {Q=Q<sub>0</sub> ∪ (q=?)}) ] </pre>	<pre> <b>agent</b>(id<sub>1</sub>, <b>IS</b>, <b>Proc</b><sup>(m)</sup>, <b>K</b><sup>(s)</sup> ∪ {Q=Q<sub>0</sub>}) [ newq=next_question(Q) then request(newq) ⇒ <b>agent</b>(id<sub>2</sub>, IP) then <b>agent</b>(id<sub>1</sub>, IS, ∅, {Q=Q<sub>0</sub> ∪ (newq=?)}) ] </pre>
<pre> or [a=checktime() then null] </pre>	<pre> or [inform(q, a) ⇐ <b>agent</b>(id<sub>2</sub>, IP) then <b>agent</b>(id<sub>1</sub>, IS, ∅, {Q=Q<sub>0</sub> ∪ (q=a)}) ] </pre>
<pre> or [ (q,a)=get_answer(Q) then inform(q, a) ⇒ <b>agent</b>(id<sub>2</sub>, IS) then <b>agent</b>(id<sub>1</sub>,IP,∅,{Q=Q<sub>0</sub> ∪ (q=a)}) ] </pre>	<pre> or [a=checktime() then null] </pre>
<pre> or [ <b>agent</b>(id<sub>1</sub>, IP, ∅, ∅) then <b>agent</b>(id<sub>1</sub>, IP, ∅, ∅) ] </pre>	<pre> or [ <b>agent</b>(id<sub>1</sub>, IP, ∅, ∅) then <b>agent</b>(id<sub>1</sub>, IS, ∅, ∅) ] </pre>
<pre> or [ buy(prot) ⇐ <b>agent</b>(id<sub>2</sub>,IS) ] then prot </pre>	<pre> or [ prot=selling_process(t,Q) then buy(prot) ⇒ <b>agent</b>(id<sub>2</sub>,IP) ] . </pre>

**Fig. 2.** An Example of  $MAP^a$  Definition for a Dialogue in a Shop

quantity and quality of the item or business selling policies. One buying protocol is chosen during run-time, allowing a simpler and reusable protocol description.

The fundamental points of the formal definition of an EREG system that uses  $MAP^a$  language for multi agent dialogues are given below, for space reasons we can not include its complete definition.

## 4 EREG System for Modelling Agent Dialogues

Given an arbitrary definition of a scene  $S = \langle R^{(n)}, P^{(n)}, M^{(k)}, K^{(j)} \rangle$ , and following  $MAP^a$  language syntax, it can be defined an EREG system  $\Sigma = (E, \mathcal{A}_1, \dots, \mathcal{A}_n)$  that simulates its behaviour as follows.

The environment  $E = (V_E, P_E)$  represents the physical medium where information shared by all the agents is stored: roles definitions  $R^{(n)}$ , protocol definitions  $P^{(n)}$ , dialogical framework  $M^{(k)}$  and common knowledge  $K^{(j)}$ . Also through the environment agents interchange (send or receive) messages and introduce new agents invocations. Therefore the environment state has the following form:

$\omega_E = \alpha + +Roles + +Protocols + +DFramework + +ShareK$  where:

- $\alpha \in ((newagent)^* \langle \rangle_{NA,id} + + (message)^* \langle \rangle_{M,id})^+$  is the dynamic part of the environment state. It is used by the agents to introduce new messages and agent invocations. Each agent  $A_i$  with identifier  $id$  can introduce agent invocations by a rewriting rule that replaces symbol  $\langle \rangle_{NA,id}$  by the new definitions followed by  $\langle \rangle_{NA,id}$ . Messages can also be introduced with the same strategy, but replacing symbol  $\langle \rangle_{M,id}$ .
- $Roles$ ,  $Protocols$ ,  $DFramework$  and  $ShareK$  are the string representations of  $R^{(n)}$ ,  $P^{(n)}$ ,  $M^{(k)}$  and  $K^{(j)}$  respectively. They correspond to the static part of the environment state, they are defined when starting the simulation and they remain unchangeable.

In each time unit of the evolution process, agents participating in the conversation communicate through the environment sending messages and taking from it messages that they receive from other agents. Environment productions  $P_E$  are in charge of deleting in each time unit all the information that agents have added to the environment in the previous time unit. This is done with the purpose of warranty that the information agents take is new and fresh one. Whenever an agent finds in the environment information that he needs or that has been sent to him, he takes it and copy it in his state. Strings *Roles*, *Protocols*, *DFramework* and *ShareK* are never deleted from  $\omega_E$ . The same happens with symbols  $\langle \rangle_{NA,id}$  and  $\langle \rangle_{M,id}$ , needed by agents to introduce new agent instances or to send messages.

Classes  $\mathcal{A}_i$ ,  $1 \leq i \leq n$ , are defined: one per each role  $role_i \in R^{(n)}$ . And every agent  $A_i = (V_i \cup \{\sqcup\}, P_i, R_i, \varphi_i, \psi_i)$  that belongs to class  $\mathcal{A}_i$  behaves as an agent playing role  $role_i$ . The definition of a scene  $S$  is completed with a list of agent invocations that start the scene. Dynamically during execution time active agents introduce new instances of agents. The same counts for the model given here: to start the simulation of the scenario  $S$ , initial environmental state  $\omega_{E_0}$  has to contain the initial agents invocations. And besides each of the  $n$  classes  $\mathcal{A}_i$  have to be defined with initial state  $W_{i_0} = \{\langle Parent \rangle\}$ . If during evolution time agents invocations are introduced in  $\omega_E$ , an agent from class  $\mathcal{A}_i$  with state  $\omega_i = \alpha \langle Parent \rangle$  performs a reproductive rule. Its state changes to  $\omega_i = \alpha$  and a new agent  $A_i$  in  $\mathcal{A}_i$  is created with initial state:

$$\begin{aligned} \omega_i = & Identifier + +IMessages + +OpClause + +CState + +UTerms \\ & + +AgentK + +ShareK + +AgentProc \langle Parent \rangle \end{aligned}$$

where:

- *Identifier* is the string representation of the agent identifier *id*.
- $IMessages \in \{\langle \rho((value, type)^{(k)}, sender) \rangle_{IM} \mid sender = [id_j, c] \wedge 1 \leq c \leq n\}$  corresponds to the messages received by the agent but not processed. So, initially  $IMessages = \langle \rangle_{IM}$ .
- $OpClause = \langle \downarrow oc \rangle_{OP}$  is the string representation of the operational clause *oc* assigned to the agent, where symbol  $\downarrow$  is used to mark the current operator. During execution time  $OpClause = \langle \alpha \downarrow op_x \beta \rangle_{OP}$  and symbol  $\downarrow$  points to the current operator  $op_x$ . When  $OpClause = \langle \alpha \downarrow \rangle_{OP}$  the agent is considered inactive, it has no more operations to perform.
- $CState \in \{\langle Close \rangle, \langle Open \rangle, \langle Inactive \rangle\}$  is used to know the current state of the agent and indicates respectively if the operator  $op_x$  in  $\langle \alpha \downarrow op_x \beta \rangle_{OP}$  has been applied, not applied or if the agent is inactive ( $OpClause = \langle \alpha \downarrow \rangle_{OP}$ ). An agent is initially created with  $CState = \langle Close \rangle$  and  $OpClause = \langle \downarrow oc \rangle_{OP}$ .
- *UTerms*, *AgentK*, *ShareK* and *AgentProc* represent the pairs of matchings between formal and real parameters used by the agent during the active life in the scene, the set of axioms corresponding to the agent private knowledge, the set of axioms common to all the agents and the set of decision procedures that the agent can execute.

With respect to mappings  $\varphi_i$  and  $\psi_i$ , we are warned of their possible complexity, so we restrict them to computable mappings. We consider  $\varphi_i : V_E^* \longrightarrow 2^{P_i}$  as a computable mapping in charge of:

- Generating the corresponding new agent instances in class  $\mathcal{A}_i$  when finding in  $\omega_E$  agents invocations with role  $role_i$  introduced by agents participating in the conversation. This is achieved by the used of synchronous reproductive rules, as described above.
- Performing the delivery of messages introduced in  $\omega_E$  by agents participating in the scene. When detecting in the environment state the presence of messages sent to the agent  $A_i$  in class  $\mathcal{A}_i$  with identifier  $id$ ,  $\varphi_i$  adds the messages to the sequence of messages received but not processed by agent  $A_i$ . But first it is checked that the received messages are valid, i.e. the performatives are from the dialogical framework  $D\text{Framework}$  in  $\omega_E$  and that there are matchings between the types of the real parameters and the formal parameters of the performatives.

$\psi_i : V_i^+ \longrightarrow 2^{R_i}$  is in charge of:

- Simulating the application of operators *then*, *or* and *the replacement of a protocol variable for its corresponding operational clause*.  $\psi_i$  also simulates actions corresponding to the performance of *decision procedures*, the *sending of a message* and the *processing of a received message*. According to the substrings  $OpClause = \langle \alpha \downarrow op_x \beta \rangle_{OP}$  and  $CState \in \{ \langle Close \rangle, \langle Open \rangle, \langle Inactive \rangle \}$  from  $\omega_i$  mapping  $\psi_i$  determines the current operator and tries to apply it.
  - If  $CState = \langle Close \rangle$ ,  $\psi_i$  interprets that marker  $\downarrow$  is pointing an operator already executed and that it should try to apply next operator in  $\langle \alpha \downarrow op_x \beta \rangle_{OP}$ . If next operator  $op_y$  can be applied, mapping  $\psi_i$  selects from  $R_i$  a set of rules according to the type of  $op_y$ . For example, if  $op_y = (\rho(\phi_1^{(k)})) \implies agent(\phi_2^{(s)})$ ,  $\psi_i$  tries to simulate the sending of a message. It replaces formal parameters  $\phi_1^{(k)}$  and  $\phi_2^{(s)}$  for matching real parameters  $p_1^{(k)}$  and  $(id_h, j)$  according to  $UTerms$ .

The new message is introduced in the conversational context replacing symbol  $\langle \rangle_{M,id}$  by string  $\langle \rho(p_1^{(k)}), [id, i], [id_h, j] \rangle_{M,id} \langle \rangle_{M,id}$ , where  $[id, i]$  represents the sender and  $[id_h, j]$  denotes the receiver. Besides the simulation of the agent participation in the conversational space, mapping  $\psi_i$  models the change in the agent mental state. It rewrites the string  $OpClause$  to make symbol  $\downarrow$  point  $op_y$ , to denote that  $op_y$  has already been performed. But in case next operator  $op_y$  cannot be applied because of absence of matchings between formal and real parameters in  $UTerms$ , mapping  $\psi_i$  changes current state to open and actualizes the current operator.

If there are no more operators to apply,  $OpClause = \langle \alpha \downarrow \rangle_{OP}$ , then the state of the agent has to be set as inactive. Mapping  $\psi_i$  selects from  $R_i$  the set of rules:  $\{ \langle Close \rangle \rightarrow \langle Inactive \rangle \}$ . In the next derivation

step, the system detects that the agent is inactive and deletes all the symbols from its state  $\omega_i$ , except the symbol  $\langle Parent \rangle$ . If the agent was in charge of creating new instances of agents of the same class ( $\omega_i = \beta \langle Parent \rangle$ ) it is not considered any more an active agent but keeps the duty of procreator until it performs the first introduction of new agent instances. As it was described before, the creation of a new agent of the class leaves the agent state in an empty one ( $\omega_i = \lambda$ ). According to the definition of REG system, an agent with state  $\omega_i = \lambda$  is inactive and the system eliminates it in the next derivation step.

- If  $CState = \langle Open \rangle$  and  $OpClause = \langle \alpha \downarrow op_x \beta \rangle_{OP}$ ,  $\psi_i$  interprets that marker  $\downarrow$  is pointing an operator  $op_x$  that has not yet been executed and tries to apply it.

If  $OpClause = \langle \alpha (\downarrow op_x \text{ or } op_y) \beta \rangle_{OP}$  or  $OpClause = \langle \alpha (op_y \text{ or } \downarrow op_x) \beta \rangle_{OP}$ ,  $\psi_i$  makes some kind of backtracking. It changes the current state to close and rewrites  $OpClause$  to  $\langle \alpha (op_x \text{ or } \downarrow op_y) \beta \rangle_{OP}$  or  $\langle \alpha (\downarrow op_y \text{ or } op_x) \beta \rangle_{OP}$  respectively. In this way a situation of deadlock for agent  $A_i$  is avoided, allowing to try with the other term of the *or* operator in next derivation step.

Briefly we can summarize main features of mappings  $\varphi_i$  and  $\psi_i$  in the following way:

1. Mapping  $\varphi_i$  is responsible for delivering messages to the corresponding agents and to create new agent instances, behaving in a fix and deterministic way.
2. Mapping  $\psi_i$  is used to model agent undeterministic and unpredictable behaviour. It simulates the execution of the agent protocol. It is in charge of implementing the agent participation in the dialogue and keeping its mental state updated. Although protocols define controlled and predictable interactions they do not restrict agent autonomy, which is achieved by agent decision procedures embedded in  $\psi_i$ . Moreover agent behaviour can be decided by the dynamic interchange of messages, as it was shown in Figure 2, being mapping  $\psi_i$  responsible of performing the replacement of protocol variables for executable protocols.

## 5 Discussion

The model we present here turns out to be appropriate to simulate multi agent systems dialogues, but our future intention is to refine it to describe human-computer conversations. In [4,7,8,2] similar grammar systems variants have been presented to perform human dialogue, but no implementation details were given. A definition of co-operating distributed grammar system with memories has been introduced in [1] with the purpose of defining human-machine interfaces. Some differences can be recognized between that model and ours:

- There  $n+1 \geq 1$  grammars  $G_i$  are defined, one grammar per speaker plus one grammar for modelling the conversational environment. The number of active speakers can vary during the conversation but the maximum number of

active agents is bounded by the  $n$  grammars introduced during design time. In our model  $m \geq 0$  classes  $\mathcal{A}_i$  are defined, one per conversational role plus the conversational environment  $E$ . The number of active speakers can also vary during the conversation but there is no restriction over the maximum number of speakers per role. The only restriction is over the number of roles (class definitions) agents can play in the scene, given by  $m$ . In [1] an example of definition for a two-party sale situation is given and it is explained that in case of more speakers model has to be rewritten. While for our framework any  $n$ -party sale conversation can be modelled using the protocol definition introduced in Figure 2. Our model definition is independent of the speakers number, only environment state  $\omega_{E_0}$  has to be initialised according to the number  $n$  of initial speakers.

- In [1] agents are defined as semi-conditional grammars  $G_i = (N_i, T_i, P_i)$  with rewriting rules  $P_i$  for describing their behaviour. While for us the behaviour of agent  $A_i$  from role class  $\mathcal{A}_j$  is given by string state  $\omega_{ji}$ . The limitations of the use of rewriting rules for defining agent protocols are clear: they are statically defined during the definition of the grammar system and they remain fix during hole simulation allowing to model only fixed state-based conversational spaces. In contrast with this, describing protocols as processes stored in strings allows to dynamically change their content during the execution. This enlarge the number of scenes to be described to those where it is impossible or impractical to define the protocol beforehand. Processes provide a more natural and intuitive way of reasoning: they make easier to identify the parts of the protocol that depend on interaction from those that do not. Moreover processes allow to apply useful strategies from the computer science paradigm rarely used in formal language frameworks. Like embedding of protocols and, due to parameterised agents definitions, reuse of components and recursive calls.

## 6 Conclusions

In this paper we introduce the formal definition of an extension of Reproductive Eco-Grammar System. This framework seems to be appropriate to simulate multi agent conversational scenes defined following  $MAP^a$  syntax. We are currently working on restricting our model to simulate human-computer conversations.  $MAP^a$  protocols are used for efficient message interchange allowing an arbitrary number  $m \geq 0$  of agents to speak simultaneously or an agent to listen  $n \geq 0$  messages at the same time. We want to limit our framework to warranty turn-taking, one agent talking per time, recovery mechanisms in case of more than one agent talking simultaneously, and other features characteristics of human dialogue.

With respect to future work it can be interesting to explore how the inclusion of the  $MAP^a$  language in EREG systems can benefit grammar system field. Multi-agent paradigm offers a powerful set of metaphors, concepts and techniques for conceptualizing, designing, implementing and verifying complex distributed systems that maybe can be adopted by the grammatical field. It is also worth to try the opposite direction: to prove that  $MAP^a$  language can



benefit from this theoretical tool. As an example we can try to find out classes of problems that  $MAP^a$  language can solve under fix restrictions.

## References

1. Aydin, S., Jürgensen, H., Robbins, L.E.: Dialogues as Co-Operating Grammars. *Journal of Automata, Languages and Combinatorics*, **6(4)** (2001) 395-410.
2. Bel-Enguix, G., Jiménez-López, M.D.: Artificial Life for Natural Language Processing. In Caprere, M., Freitas, A., Bentley, P., Johnson, C., Timmis, J. (eds.), *Advances in Artificial Life*, LNAI 3630. Springer, Berlin (2005) 765-774.
3. Csuhaj-Varjú, E., Dassow, J., Kelemen, J., Păun, Gh.: *Grammar Systems. A Grammatical Approach to Distribution and Cooperation*. Gordon and Breach, London (1994).
4. Csuhaj-Varjú, E., Jiménez-López, M.D., Martín Vide, C.: "Pragmatic Eco-Rewriting Systems": Pragmatics and Eco-Rewriting Systems. In Păun, Gh. & Salomaa, A. (eds.), *Grammatical Models of Multi-Agent Systems*. Gordon and Breach, London (1999) 262-283.
5. Csuhaj-Varjú, E., Kelemen, J., Kelemenová, A., Păun, Gh.: Eco-Grammar Systems: A Grammatical Framework for Life-Like Interactions. *Artificial Life*, **3, 1** (1996) 1-28.
6. Grando, M.A. , Walton, C.: The  $MAP^a$  Language of Agent Dialogues. In *Proceedings AAMAS 2006, Fifth International Joint Conference on Autonomous Agents and Multiagent Systems*, Japan (2006).
7. Jiménez-López, M.D.: Dialogue Modelling with Formal Languages. In Spoto, F., Scollo, G., Nijholt, A. (eds.), *Algebraic Methods in Language Processing*. TWLT 21, University of Twente (2003) 207-221.
8. Jiménez-López, M.D.: *Using Grammar Systems*. GRLMC Report, Rovira i Virgili University, Tarragona (2002).
9. Rozenberg, G., Salomaa, A. (eds.): *Handbook of Formal Languages*. Springer, Berlin (1997).
10. Walton, C.: Multi-agent Dialogue Protocols. In *Proceedings of the Eighth International Symposium on Artificial Intelligence and Mathematics*. Fort Lauderdale, Florida (2004).

# A Calculus for MAS Interaction Protocol

Hongbing Chen, Qun Yang, and Manwu Xu

State Key Laboratory for Novel Software Technology  
Nanjing University, Nanjing 210093, China  
21-suns@163.com, mwxu@nju.edu.cn

**Abstract.** Formal description and verification of the interaction protocols between agents is a very valuable research in development of MAS. In the paper we have defined a calculus for description interaction protocols of MAS based on dialogue. The calculus is founded on process algebra and is designed to be independent of any particular model of rational agency. This makes the verification applicable to heterogeneous agent systems. With the state of session environment and formal semantics of calculus, we can verify some properties of session protocols, e.g. termination, deadlock. Our approach does not suffer from the semantic verification problem because the states of the session dialogue are defined in the protocol itself, and it is straightforward to verify that an agent is or not acting in accordance with the protocol and does not suffer from state-space explosion.

**Keywords:** Interaction protocol, Multi-agent, Dialogue protocol,  $\pi$ -Calculus.

## 1 Introduction

Semantics of communication acts between agents in Multi-agent System(MAS) is one of important research fields of MAS. So research of interaction model and its semantics is very valuable. Many researches are based on the theories of speech acts[1]. Social interactions, such as cooperation, coordination and negotiation, are enacted through a variety of Agent Communication Languages(ACLs) and interaction protocols. An ACL(KQML, FIPA ACL[2], etc) specifies the individual communicative acts, typically as classes of asynchronous messages modelled on speech acts theories. The semantic specification for FIPA ACL is expressed using a BDI logic, derived from[3]. While this specification is informative, it has been criticized on various grounds[4,[5]. Consequently, while the published protocol specifications may well be helpful for comprehension, in practice they are inadequate for prescribing the individual messages of an implementation. For lack of precision arises from the inherent inexpressiveness of a diagrammatic representation such as Petri-nets and AUML[6], and from a focus on a conceptual level of discussion using informal language rather than formal logic. Three main approaches have been proposed to model communication between agents in general and to define a semantics for ACLs. These approaches are the mental approach, the social approach, and the argumentative approach.

In the mental approach, so-called BDI models are used to model conversations and to define a formal semantics of speech acts. It was used by [3,6,7] to define a formal

semantics of KQML and FIPA ACL. However, these semantics have been criticized for not being verifiable because one can't verify whether the agents' behaviors match their private mental states [5,8].

In order to avoid the problems associated with the BDI models, and thereby to express a greater range of dialogue types, a number of alternative semantics for expressing rational agency have been proposed. There are two approaches that have received the most attention. One is semantics based on social commitments, and the other is semantics based on dialogue games[10]. The key concept of the social commitment model is the establishment of shared commitments between agents. It is based on social commitments that are thought of as social and compulsory notions. Social commitments are commitments towards the other members of a community. They differ from the agent's internal psychological commitments, which capture the persistence of intentions as specified in the rational interaction theory. As a social notion, commitments are a base for a normative framework that makes it possible to model the agents' behavior.

The argumentative approach was proposed by [12] as a method for modeling dialogue. It is based upon an argumentation system where the agents' reasoning capabilities are often linked to their abilities to argue. They are mainly based on the agent's ability to establish a link between different facts, to determine if a fact is acceptable, to decide which arguments support which facts, etc. The approach relies upon the formal dialectics introduced by [13].

Despite all this research focused on modeling dialogue and semantic issues, there is an additional problem of verification of interaction protocols. A MAS defines a complex concurrent system of communicating agents. Concurrency introduces non-determinism into the system, which gives rise to a large number of potential problems, such as fairness, and deadlocks. Traditional debugging and simulation techniques cannot readily explore all of the possible behaviors of such systems, and therefore significant problems can remain undiscovered. The detection of problems in these systems is typically accomplished through the use of formal verification techniques such as theorem proving and model checking. The model checking technique is appealing as it is an automated process, though it is limited to finite-state systems. One of the main issues in the verification of software systems using model checking techniques is the state-space explosion problem. The exhaustive nature of model checking means that the state space can rapidly grow beyond the available resources as the size of the model increases. Thus, in order to successfully check a system it is necessary that the model is as small as possible.

The aim of this paper is to describe and verify interaction protocol formally. We do not adopt a specific semantics of rational agency. Our belief is that in a truly heterogeneous agent system we cannot constrain the agents to any particular model. Instead, we define a model of dialogue that separates the rational processes and interactions from the actual dialogue itself. We define a simple but powerful calculus of Multi-agent Interaction Protocol(MIP) as an alternative to the state-chart representation of protocols. Our formalism allows the definition of infinite-state dialogue and the mechanical processing of the resulting dialogue protocols. MIP protocols contain only representation of the communicative processes between agents and the resulting models are therefore significantly simpler. This approach does not suffer from the semantic verification problem because the states of the dialogue are defined in the protocol itself.

The remainder of this paper is structured in the following way. The syntax of Multi-agent Interaction Protocol Calculus and example protocol of calculus are defined in Section 2. In section 3 we introduce an environment of session and present the semantics of MIP-Calculus. How to verify the properties of session protocols is discussed in section 4. Finally, in section 5 we provide the conclusion and future work.

## 2 MIP-Calculus

Among many formal tools, process algebras have often been proposed as a useful paradigm for the specification and analysis of the software behavioral properties. We have adapted  $\pi$ -calculus[21] into a suitable process algebra, called Multi-agent Interaction Protocol Calculus (MIP-Calculus). Agents are defined as processes of this calculus. The MIP-Calculus is a simple dialogue protocol language that is an extension of basic  $\pi$ -calculus with some primitives. The underlying semantics of our calculus is derived from labeled transition system.

### 2.1 The Syntax of MIP-Calculus

In our calculus the concepts of session and role are crucial. A session is dialogue based environment and can be thought of as a bounded space in which a group of agents interact on a task. We assume that a session places barrier conditions on the agents, such that a session cannot begin until all the agents are present, and the agents cannot leave the session until the dialogue is complete. The role of an agent is used to identify some task, behavior, responsibility or function that should be performed within the interaction protocol. A role describes the expected behavior and properties of an agent. A role also prescribes the procedures and rules in an session. We consider an agent role as a specification of process that is expected to be available to agents playing that role. A role also identifies capabilities of which the agent must provide.

The syntax of MIP-Calculus is presented in Fig. 1. A session specification is a two triples with a unique name  $SName$ . The number of roles is notated by  $n$  ( $n \geq 2$ ).  $R^n$  is the name set of roles, and  $P^n$  is agent protocols each of which defines a process. Agent has a fixed role for the duration of the protocol, and is individually identified by unique name  $a$ . Protocols are constructed from actions which control the flow of the protocol.

The action expression  $0$  corresponds to idle action, while  $\tau.E$  represents a behavior of silent action and then behavior  $E$ . The silent action denotes an internal computational step that an agent can perform independently of its environment. Interaction between agents is performed by the exchange of messages  $Msg$  which contain performatives.  $S!(Msg, m).E$  is an output action sending message through channel  $S$  then acting as  $E$ . As for receiving message, we don't have primitive action. By action of *waitfor*, if the message queue of agent is not empty then the agent executes the action  $Match(Msg)$  and decide to some other action dependent on the message type of performative content. The interface between the protocol and the rational process of the agent is achieved through the invocation of decision procedures  $Match(Msg)$ .

The actions in the protocol are sequenced by the  $(E1 ; E2)$  which acts  $E1$  followed by  $E2$ , unless  $E1$  involved an action which failed. The parallel composition of two behavioral expressions  $(E1 \parallel E2)$  can execute, in any order, the action that  $E1$  or  $E2$  is

ready to execute, and the nondeterministic choice ( $E1 + E2$ ) can either behave as  $E1$  or as  $E2$ , exclusively. The basic term is one of these: variable  $v$ , agent name  $a$ , role name  $r$ , constant  $c$ , or wild-card  $_$ .

$S ::= SName [R^n, P^n]$	(Session Protocol)
$P ::= agent(a, r, E)$	(Agent Protocol)
$E ::=$	(Process)
$!0$	(Idle State)
$! \tau.E$	(Silent $\tau$ )
$! S ! (Msg, m).E$	(Send message)
$! Match(Msg)(MsgType_1 \rightarrow E_1; \dots ; MsgType_k \rightarrow E_k).E$	(Decision procedure)
$! waitfor E_1 timeout (t) E_2$	(Iteration)
$! exit (a)$	(Exit the session)
$! (A\{E\} = A\{q(E)\})$	(Recursive behaviour $E, E = q(E)$ )
$! E1 ; E2$	(Sequential)
$! E2 + E2$	(Choice)
$! E1 \parallel E2$	(Parallel)
$Msg ::= Message(MsgType, Sender, \{Receiver\}, Content, \dots)$	(Message)
$\Theta ::= v \mid a \mid r \mid c \mid _$	(Term)

Fig. 1. Syntax of MIP-Calculus

For the reason that we focus on the interaction between agents, all agents interact with each other through sending or receiving messages. So input and output action are omitted in our calculus. It is important to note that MIP is only intended to express protocols, and is not intended to be a general-purpose language for computation.

## 2.2 Example of Interaction Protocol

It is helpful to consider an example of the calculus in order to illustrate these concepts. We will now define a contract net protocol, which will illustrate the expressive power of MIP language and act as an example for protocol verification.

The initiator (Tenderer) solicits  $m$  proposals from other agents by issuing a call for proposals act (cfp), which specifies the task, as well any conditions the tenderer is placing upon the execution of the task. Tenderers receiving the call for proposals are viewed as potential contractors and are able to generate  $n$  responses. Of these,  $j$  are proposals to perform the task, specified as propose acts. The tenderer's proposal includes the preconditions that the tenderer is setting out for the task, which may be the price, time when the task will be done, etc. Alternatively,  $i = n - j$  tenderers may refuse to propose. Once the deadline passes, the tenderer evaluates the received  $j$  proposals and selects agents to perform the task; one, several or no agents may be chosen.  $i$  agents of the selected proposal(s) will be sent an accept-proposal act and the remaining  $k$  agents will receive a reject-proposal act. The proposals are binding on the tenderer, so that once the

tenderee accepts the proposal, the tenderer acquires a commitment to perform the task. Once the tenderer has completed the task, it sends a completion message to the tenderee in the form of an inform-done or a more explanatory version in the form of an inform-result. However, if the tenderer fails to complete the task, a failure message is sent.

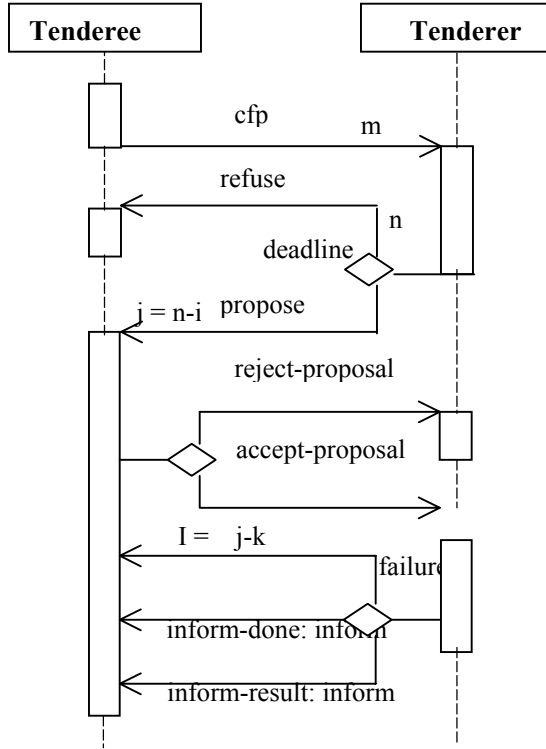


Fig. 2. FIPA Contract Net Interaction Protocol with AUML

The definition of the contract net protocol in MIP-Calculus is presented in Fig. 3. We define two roles: tenderee and tenderer. Each of these roles has a process specification which define the protocol states for the role. The session begins with an *cfp* message from the tenderee to *m* tenderers, which we denote with the message ( $Msg(cfp, A, B^m, P), m$ ). Then the tenderee is idle and wait for messages. When the waiting time *t1* is out, the tenderee deals with the received message through messages matching. If message type (performative) is *non-understood* or *refuse*, the tenderee removes the message from message queue and turn to idle state. As for all messages of *propose* of the tenderers, the tenderee enters a deliberative state, in which a decision is required. The tenderee can reject or accept a proposal. So, the tenderee accept proposals to *I* tenderers and sends message ( $Msg(accept-proposal, A, B^I, P), I$ ). At the same time, the tenderee reject proposals to *k* tenderers and sends message ( $Msg(reject-proposal, A, B^k, R), k$ ). If a *accept-proposal* is made, the tenderer deliberates

```

1 Contract-Net-Session-Protocol
2 [{Tenderee, Tenderer}, {P} :
3   Tenderee (A) =
4     { S ! ( Msg(cfp, A, Bm, P), m) .
5     waitfor
6       (Match(Msg) .
7         (MsgType = (not-understood, B, A, T) → 0 ;
8         MsgType = (refuse, Bi, A, R) → 0 ;
9         MsgType = (propose, Bj, A, P) →
10        (S ! (Msg(reject-proposal, A, Bk, R), k) |
11          S ! (Msg(accept-proposal, A, Bl, P), I) .
12        waitfor
13          Match(Msg) .
14          (MsgType = (failure, B, A, R) → exit(A) ;
15          MsgType = (inform, B, A, P:Done) → exit(A) ;)
16          (MsgType = (not-understood, B, A, T) → 0) )
17        timeout(t2)    exit(A) )
18      timeout(t1)    exit(A)
19    }
20   Tenderer (B) =
21     { waitfor
22       (Match(Msg) .
23       (MsgType = (cfp, A, B, P) →
24       (S ! (Msg(refuse, B, A, R), 1) ) .exit(B) +
25         S ! (Msg(propose, B, A, P), 1) .
26       waitfor
27         Match(Msg) .
28       (MsgType=(reject-proposal, A, B, R)→exit(B) ;
29       MsgType=(not-understood, A, B, P) →exit(B) ;
30       MsgType=(accept-proposal, A, B, P) →
31         S ! ( Msg(failure, B, A, R) →exit(B);) +
32         S ! (inform, B, A, P:Done) →exit(B);) .
33       timeout(t3)    (B) );
34       MsgType = (not-understood, A, B, P) →exit(B);)
35       timeout(t4)    exit(B)
36     }
37 ]

```

**Fig. 3.** MIP-Calculus Contract Net Interaction Protocol

further. Then, he sends a message of *inform* or *failure*. After receiving the message, the tenderee leaves the session and the protocol terminates.

### 3 Semantics of MIP-Calculus

In this section, we present a formal semantics for the MIP-Calculus. The provision of a clean and unambiguous semantics for our MIP-Calculus is a primary consideration in the design of the calculus. The purpose of the semantics is to formally describe the meaning of the different calculus constructs, such that dialogue protocols expressed in the language can be interpreted in a consistent manner. This is a failure of the formal semantics of FIPA, which is expressed by BDI logic. The FIPA semantics is an abstract description, which neglects practical aspects. Furthermore, the BDI modalities can be interpreted in a number of different ways, meaning that implementations of BDI agents have typically been ad-hoc in nature.

Semantics of MIP-Calculus is given as a Labeled Transition System(LTS), according to the seminal idea of Structured Operational Semantics[24], whose states represent processes and whose rules model how a state can evolve to another one. Transitions are labeled with information about the (observable)action that the agent exhibit, i.e. communication or silent actions. The semantics of MIP-Calculus is given by a pair of LTS. The first one, action semantics, defined in Fig.4, models the intentional behavior of the interaction patterns in isolation: executed actions are observable as labels. The second one, session semantics of Fig.5, models the semantics of interactions within a session.

The action semantics is defined up to structural congruence  $\equiv$  (+ and  $\parallel$  are associative monoidal transformation operators with 0 as neutral element and renamed terms are equivalent[19]) as standard for  $\pi$ -calculus. A action expression consisting of a prefix can execute the prefix action (rules ( $\tau$ ) and (Act)), the parallel composition of two expressions can evolve according to both the expressions (rule (Par)), with the standard condition to avoid free variable capture, while the choice of two expressions can non-deterministically evolve according to one of the two expressions (rule (Choice)). And rule (Struct) deals with structural equivalence.

$\tau.E \xrightarrow{\tau} E$	$\alpha.E \xrightarrow{\alpha} E$	(Act)
$E \xrightarrow{\alpha} E'$	$E \parallel F \xrightarrow{\alpha} E' \parallel F$	(Par)
$E \parallel F \xrightarrow{\alpha} E' \parallel F$	$bv(\alpha) \cap fv(F) = \emptyset$	
$E \xrightarrow{\alpha} E'$	$E + F \xrightarrow{\alpha} E'$	(Choice)
$E \equiv E', E' \xrightarrow{\alpha} F', F \equiv F'$	$E \xrightarrow{\alpha} F$	(Struct)

Fig. 4. Action Semantics



**Session Environment**  $\Delta ::= (SID, \prod_{i \in n} R_i, \prod_{i \in n} TR_i, MQ, \prod_{j \in m} T_j)$

**(Enter Session)**

$$\begin{array}{c}
 A : R_i \wedge A \notin R_i \wedge A \notin TR_i \\
 \prod_{i \in n} R_i = R_1\{\dots\}, \dots, R_i\{\dots\}, \dots, R_n\{\dots\} \\
 \prod_{i \in n} TR_i = R_1\{\dots\}, \dots, R_i\{\dots\}, \dots, R_n\{\dots\} \\
 P_A \xrightarrow{S!(Msg(MsgType, A, B, T))} P_A' \\
 \hline
 (\prod_{i \in n} R_i, \prod_{i \in n} TR_i, P_A) \xrightarrow{S!(Msg(MsgType, A, B, T))} (\prod_{i \in n} R'_i, \prod_{i \in n} TR'_i, P'_A) \\
 \prod_{i \in n} R'_i = R_1\{\dots\}, \dots, R_i\{\dots, A\}, \dots, R_n\{\dots\} \\
 \prod_{i \in n} TR'_i = R_1\{\dots\}, \dots, R_i\{\dots, A\}, \dots, R_n\{\dots\}
 \end{array}$$

**(Leave Session)**

$$\begin{array}{c}
 A : R_i \wedge A \in R_i \\
 \prod_{i \in n} R_i = R_1\{\dots\}, \dots, R_i\{\dots, A\}, \dots, R_n\{\dots\} \\
 P_A \xrightarrow{exit(A)} P_A' \\
 \hline
 (\prod_{i \in n} R_i, P_A) \xrightarrow{exit(A)} (\prod_{i \in n} R'_i, P'_A) \\
 \prod_{i \in n} R'_i = R_1\{\dots\}, \dots, R_i\{\dots\}, \dots, R_n\{\dots\}
 \end{array}$$

**(Match Message 1)**

$$\begin{array}{c}
 MsgType = MsgType, \neq \text{inform} \\
 P_A \xrightarrow{Match(Msg) (MsgType_1 \rightarrow E_1; MsgType_2 \rightarrow E_2; \dots; MsgType_k \rightarrow E_k)} E_i.P_A' \\
 \hline
 (MQ\{\dots, Msg\}, P_A) \xrightarrow{Match(Msg) (MsgType_1 \rightarrow E_1; \dots; MsgType_k \rightarrow E_k)} (MQ\{\dots\}, E_i.P'_A)
 \end{array}$$

**(Match Message 2)**

$$\begin{array}{c}
 MsgType = MsgType, = \text{inform} \\
 P_A \xrightarrow{Match(Msg) (MsgType_1 \rightarrow E_1; MsgType_2 \rightarrow E_2; \dots; MsgType_k \rightarrow E_k)} E_i.P_A' \\
 \hline
 (MQ\{\dots, Msg\}, \prod_{j \in m} T_j : \text{UnDone}, P_A) \xrightarrow{Match(Msg) (MsgType_1 \rightarrow E_1; \dots; MsgType_k \rightarrow E_k)} (MQ\{\dots\}, \prod_{j \in m} T_j : \text{Done}, E_i.P'_A)
 \end{array}$$

**Fig. 5.** Session Semantics

Agents are situated in some environment or other. Thus, any abstract model of agents will invariably consider (1) an environment  $\Delta$ , viewed as a set of environment states  $\Delta = \{U_1, \dots, U_n, \dots\}$  and (2) a set of actions, or environment transformer  $\text{Act} = \{\alpha_1, \dots, \alpha_n, \dots\}$ , where a transition  $U \xrightarrow{\alpha} V$  indicates a state transformation, due to  $\alpha$  being performed.

In our semantics, the session state is captured by a session environment  $\Delta$  that is defined in Fig.5. The session environment  $\Delta$  contains an n-tuple sets, comprising the session identity SID, the name set  $\prod_{i \in n} R_i$  of active agents as the role  $R_i$  in the session, the trace name set  $\prod_{i \in n} TR_i$  of agents that have entered the session as the role  $R_i$ , a message queue MQ waiting to dispose, and the state of the tasks  $\prod_{j \in m} T_j$ . There are also evaluation and reasoning environment in normal multi-agent model. We define the session environment  $\Delta$  as above for our interesting.

The semantics of message passing corresponds to reliable, buffered, non-blocking communication. Sending a message will succeed immediately if an agent matches the definition, and the message  $Msg$  will be stored in a buffer on the recipient. Sending a message will fail if no agent matches the supplied terms, and receiving a message will fail if no message matches the message template. Session semantics are illustrated as Fig. 5. ( In order to simplify transition rule, invariable parts of session environment are not listed in the rule.) The semantic of *Enter Session* is that when a agent with name  $A$  as role  $R_i$  enters into the session and sends a message to other agents, then the agent name  $A$  will be added into name set  $\prod_{i \in n} R_i$  of active agent and the trace name set  $\prod_{i \in n} TR_i$  of his role  $R_i$  if the name is not in above sets.

After finished the interaction, the agent  $A$  may execute  $exit(A)$  and leave the session, then the agent name will be removed from the set of active agent name of role  $R_i$ . This is the semantics of *Leave Session*. In our calculus receiving a message is implicit. The message will be added into the message queue  $MQ$  when the message arrives to receiver. The receiver will dispose of message through action of *Match (Msg)*. The message  $Msg$  supplied in the definition is treated as a template to be matched against any message, e.g. FIPA ACL Message Structure. The process of dealing with message is a decision procedure depending on the interior state and reasoning process of agent. If the  $MsgType(\text{performative})$  is not *inform*, the agent performs relevant action and remove the message from  $MQ$ . When the  $MsgType(\text{performative})$  is *inform*, agent does the same action as before but also changes the state of corresponding task from *UnDone* to *Done*. In our model any agent completed any task, the agent will send a message as message type of *inform* to return the result or information.

## 4 Verify the Session Protocol

In order to verify properties of interaction protocol we have some assumptions as follows:(1)Each agent has a unique name in MAS. The infrastructure of MAS must support this service; (2)A unique session ID is also assigned for every session, so each message contains the session ID explicitly. With this mechanism an agent can participate in more than one session respectively;(3)Within a session protocol, the roles of participants are complete and not redundant.

**Definition 1 (Session initial state).** When a session is ready to begin, the current session has a identity  $SID$ , the name set of active agent, the name set of traced agent for all roles and message queue waiting to dispose are null and the state of tasks is *UnDone*.

$$\Delta ::= (SID, \prod_{i \in n} R_i = \{\phi\}, \prod_{i \in n} TR_i = \{\phi\}, MQ = \{\phi\}, \prod_{j \in m} T_j : UnDone)$$

When the session is ready to begin, none active agent has entered the session. Therefore, the name set of active agent for all roles is empty, i.e.  $\prod_{i \in n} R_i = \{\phi\}$ . The name set of traced agent is null also.

**Definition 2 (Session termination).** After session is started, if there is none active agent in the session ( $\prod_{i \in n} R_i = \{\phi\}$ ) then session is terminated.

When the session is terminated, we will also take into account some other things. For example, the tasks of the session are completed or not, each role of session has been executed at least once, etc.

**Proposition 1 (Session normal termination).** A session is normal terminated, if and only if the current session environment provided that (1) active agent names of all roles and message queue waiting to deal with are null and the state of all tasks is *Done*, and (2) race of agent names of all roles are not null.

$$\Delta ::= (SID, \prod_{i \in n} R_i = \{\phi\}, \prod_{i \in n} TR_i = \{\neg\phi\}, MQ = \{\phi\}, \prod_{j \in m} T_j : Done)$$

Proof

- 1) When the session is started, the initiator agent( $a, R_i, P_i$ ) will enter the session and send a message to other agents. With the session semantics of *Enter Session*, name  $a$  of the agent( $a, R_i, P_i$ ) will be added to the sets of  $R_i\{a\}$  and  $TR_i\{a\}$ . After finished the interaction, agent( $a, R_i, P_i$ ) must execute *exit(a)* and leave the session, then the agent name will be removed from the set  $R_i\{a\}$ . The session is normal terminated that means all the agents must leave the session correctly, so  $\prod_{i \in n} R_i = \{\phi\}$  ( $n$  is the number of roles in the session).
- 2) With the assumption of 3), the session can correctly play and complete the tasks only that each role of agent has at least one agent instance participated the session. With the session semantics of *Enter Session*, we can conclude that  $\prod_{i \in n} TR_i = \{\neg\phi\}$ .
- 3) With the semantics of *Match Message*, all message within the session will be disposed by the receiver. So when the session is terminated that  $MQ = \{\phi\}$ .
- 4) When the session is correctly terminated,  $\prod_{j \in m} T_j : Done$  is intuitive. ■

**Definition 3 (Session abnormal termination).** If a session is terminated, but it is not normal termination. We call the session is abnormal termination.

**Proposition 2 (Session non-termination).** After the limited time, a session can't terminate, i.e.  $\exists i, R_i = \{\neg\phi\}$ . The session is non-termination. The proof is simple and intuitive.

Our verification of interaction protocol is focused on the termination. This is an important consideration in the design of interaction protocols. Non-termination can occur as a result of many different issues such as deadlocks, live-locks, infinite

recursion, and message errors. We also want to ensure that protocols do not simply terminate due to failure within the session. With the proposition 1, the termination condition is the most straightforward to validate.

## 5 Conclusions and Future Work

In the paper we have defined a calculus for description interaction protocols of MAS based on dialogue. Our calculus of multi-agent dialogue protocols fills an essential gap between the low-level communication and high-level reasoning processes in MAS. The calculus is founded on process calculus and is expressive enough to describe a large range of agent protocols.

Our calculus is designed to be independent of any particular model of rational agency. This makes the verification applicable to heterogeneous agent systems. With the definition of session environment and formal semantics of calculus, we can verify some properties of session protocols. This approach does not suffer from the semantic verification problem because the state of the dialogue is defined in the protocol itself, and it is straightforward to verify that an agent is acting in accordance with the protocol. We believe this will allow us to overcome the problems of the BDI model highlighted in the introduction, and will yield models that do not suffer from state-space explosion.

Nonetheless, we recognize that the BDI model is still of significant importance to the agent community. To address this issue, we are currently defining a system which translates FIPA ACL specifications into MIP protocols. Our current research is how to extend the range of properties of dialogue protocols that can be verified by our calculus.

## References

1. Searle, J. R.: *Speech acts: An Essay in the Philosophy of Language*. Cambridge University Press (1969)
2. FIPA: FIPA Communicative Act Library Specification(SC00037). Foundation for Intelligent Physical Agents, <http://www.fipa.org/spec> (2002)
3. Sadek, D.: A Study in the Logic of Intentions. 3rd Conf. on Principles of Knowledge Representation and Reasoning(1992)462-473
4. Pitt, J., Mamdani, A.: Communication Protocols in MAS. Workshop On Specifying and Implementing Conversation Policies. (1999) 39-48
5. Wooldridge, M.: Semantic Issues in the Verification of Agent Communication Languages. *Journal of Autonomous Agents and MAS* (2000)3(1) 9-31,
6. Labrou, Y.: *Semantics for an Agent Communication Language*. Ph.D Thesis, University of Maryland, USA,(1997)
7. Labrou, Y., Finin, T.: *Semantics and Conversation for an Agent Communication Language*. Readings in Agents, Morgan Kaufman Publisher, (1998)235-242
8. Dignum, F., Greaves, M.: Issues in Agent Communication: an Introduction. F. Dignum and M. Greaves(eds.): *Issues in Agent Communication*(2000)1-16
9. Singh, M.P.: *Agent Communication Languages: Rethinking the Principles*. IEEE Computer, (1998)40-47
10. Maudet, N., Chaib-draa, B.: Commitment-based and Dialogue-game based Protocols: News Trends in Agent Communication Language. *The Knowledge Engineering Review*(2002) 17(2) 157-179

11. Singh, M.P.: A Social Semantics for Agent Communication Language. F. Dignum and M. Greaves(eds.). *Issues in Agent Communication*(2000)31-45
12. Amgoud, L., Maudet, N., Parsons, N.: Modelling Dialogues Using Argumentation. *Proceeding of the 4th International Conference on MAS*(2000)31-38
13. MacKenzie, J.: Question-begging in Non-cumulative Systems. *Journal of Philosophical Logic*(1979)(8)117-133
14. McBurney, P., Parsons, S.: Games that Agents Play: A formal Framework for Dialogues between Autonomous Agents. *Journal of Logic, Language and Information*(2002)11(3) 315-334
15. Wooldridge, M., Fisher, M., Huget, M. P., Parsons, S.: Model Checking Multiagent Systems with MABLE. In *Proceedings of AAMAS-02*, Bologna, Italy, July
16. Greaves, M., Holmback, H., Bradshaw, J.: What is a Conversation Policy?. In *Proceedings of Agents '99*, Seattle, Washington, May
17. Labrou, Y., Finin, T.: Semantics and Conversations for an Agent Communication Language. In *Proceedings of IJCAI-97*(1997)584-591, Nagoya, Japan, August
18. FIPA: FIPA ACL Message Structure Specification(SC00061G). Foundation for Intelligent Physical Agents, <http://www.fipa.org/spec>
19. Milner, R.: *Communication and Concurrency*. Prentice Hall(1989)
20. Gaspari, M., Motta, E.: Symbol-level Requirements for Agent-level Programming. In Cohn, A. (ed.): *ECAI94 the 11th European Conference on Artificial Intelligence*(1994)364–368
21. Milner, R., Parrow, J., Walker, D.: A Calculus of Mobile Processes, Parts I and II. *Information and Computation*(1992)100(1) 1-40 and 41-77
22. Goltz, U., Gorrieri, R., Rensink, A.: Comparing Syntactic and Semantic Action Refinement. *Information and Computation*(1996)125(2) 118–143

# Synthesizing Stigmergy for Multi Agent Systems

Grant Blaise O'Reilly<sup>1</sup> and Elizabeth Ehlers<sup>2</sup>

<sup>1</sup> Academy of Information Technology, University of Johannesburg, Auckland Park,  
Johannesburg, South Africa  
oreill\_g@mtn.co.za

<sup>2</sup> Academy of Information Technology, University of Johannesburg, Auckland Park,  
Johannesburg, South Africa  
eme@na.rau.ac.za

**Abstract.** In order to synthesize stigmergy a model needs to be created that allows a collective of agents to achieve global results through local interactions in some environment. This locality of interactions between the agents and between the agent and the environment allows for the distribution of the entire system without any centralization. Stigmergy is found among social insects in nature. These natural systems show remarkable flexibility, robustness and self-organisation. These characteristics are sort after in modern software systems. Utilizing stigmergy in an artificial system allows agents to interact with one another and with the general topology in a non-centralized manner, thus giving rise to a collective solution when solving of certain tasks. Even though the agents are localized their interaction with the stigmergy layer allows other agents to be affected by the interactions. The methodology of mimicking stigmergy into a software system will be described and a description of the model used to synthesize stigmergy will be given. The potential utilization of stigmergy by software agents to interact with each other and to solve certain tasks collectively is also demonstrated.

## 1 Introduction

The indirect communication among individuals through the environment in which they exist is known as stigmergy. Stigmergy can be found in social insects and the collective movement phenomena such as flocking, herding and shoaling found in different animal societies.

Natural systems such as social insects that utilise stigmergy have been very successful almost everywhere in the ecosphere. This may be attributed to three characteristics, namely flexibility, robustness and self-organization [1]. These characteristics can be identified in enterprise organizations today. Enterprise applications need to be robust, flexible and self-regulating in order to adapt to the rapid growth and constant needs of a changing business environment [2],[3].

Eighty percent of all software projects are deemed failures due to the software systems inability to adapt to a changing business environment [2],[3]. Stigmergy may have a major role to play in the future of enterprise organizations as it may be used to eliminate problems of flexibility, robustness and self-regulation lacking in current enterprise applications.

## 2 Stigmergy

Natural systems have been evolving for billions of years and exhibit phenomenal flexibility, adaptiveness, robustness and self-organization. From this immense amount of experience displayed in natural systems it would make sense to use nature as a mentor in the quest to develop software systems with similar characteristics. In this section the natural building blocks of stigmergy will be laid out. These natural characteristics of stigmergy need to be described distinctly as it is the base of the proposed model.

In a natural complex adaptive system (CAS) such as a colony of social insects, self-organization and flexibility in the collective requires interaction among the individuals in the colony [4],[5]. Interaction can be direct or indirect. Direct interactions are usually via the physical senses of the individuals while indirect interaction is via the environment in which the individuals exist. Indirect interaction through the environment arises when an individual modifies the environment and the other individuals in the collective respond to the changing environment. Interaction among individuals via the environment in which they exist is an example of stigmergy [4].

Stigmergy can also be seen as a mechanism where by the coordination and regulation of individuals in a collective is not governed by the individuals but instead by the activities that are being performed. When a stigmergy mechanism is in place the individuals do not direct their activities, the activities actually direct them [4]. Stigmergy provides a means that relates the behavior of an individual to the collective's behavior [4]. For example the perturbations induced in the environment by an individual's behavior may in turn influence the behavior of the other individuals in the collective.

In natural systems such as insect colonies when an external perturbation occurs the stigmergy mechanism allows the insects to respond to the changing environment as if it were a perturbation caused by the colonies activities in the environment. This type of flexibility would be priceless in a modern enterprise system as it would mean enterprise systems would be able to with stand external perturbations with out any re-programming needed. Self-organization in many of the activities in a natural system results in emergent behavior such as pheromone trail creation, recruitment for a task, grouping of similar objects and regulation of task management. Stigmergy forms part of the driving mechanism behind self-organization and resilience displayed in some natural occurring complex adaptive systems [1].

## 3 Artificial Stigmergy in Business Architectures

Networks today are becoming a universal means through which users can access applications via personal computers, hand-held devices and wireless devices. Many day-to-day transactions that are taking place through distributed applications depend on the reliability, availability and security of these distributed applications. Due to the ever increasing demands that are being placed on networks and distributed systems the complexity in distributed systems is increasing drastically. These modern systems have the characteristics of been extremely complex with unpredictable interactions among their distributed components. The increasing high dynamism of modern systems results in system behavior that is impossible to formally reason about. Traditional means for design and developing distributed systems is starting to show

their inadequacies in dealing with the dynamism of the modern distributed applications [6]. The reason for this is that most distributed applications are designed in an inflexible and centralized manner. A centralized (top-down) design is often due to the ease of system management. However distributed systems designed in a centralized manner are not adaptive to changing environments.

The evolution of distributed systems is nearing a point of complexity that puts it far from the reach of traditional techniques utilized for designing and managing these systems [6],[7]. A framework is required that will be able to handle the continuous growth of complexity as well as enable the design and development of robust, flexible, self-organizing and self-adaptive distributed system.

Research fields [6],[7] in business architectures have suggested that the future developments of modern distributed enterprise systems will have to display characteristics such as resilience, adaptability and self-organization. The global properties of complex adaptive systems such as resilience, adaption and self-organization are exactly the desired characteristics for enterprise applications.

What makes stigmergy an attractive topic for enterprise applications is that it seems to facilitate self-organization and resilience in complex adaptive systems such as social insect colonies. If a stigmergy framework could be utilized to facilitate a complex adaptive system that allows self-organization and resilience to emerge in distributed and enterprise applications a new paradigm in designing and developing systems would have been started.

## 4 The Artificial Collective Engine Utilising Stigmergy

The Artificial Collective Engine Utilising Stigmergy (ACEUS) framework is an attempt to model an artificial complex adaptive system that can support the synthesizing of stigmergy. Stigmergy has been an inspiration for a number of interesting approaches to multi agent systems [8],[9]. From these particular studies it can be seen that the agent model is rather simple and that the environment is very rudimentary. The stigmergy synthesis in the ACEUS model attempts to exhibit cognitive agents, where there is a high level of knowledge representation and goal representation in the agents via the agent's ontology and rules engine. Thus, allowing the ACEUS model the ability to construct multi agent stigmergic mechanisms that can coordinate complex activities. ACEUS is modeled around a Complex Adaptive Systems (CAS). CAS have been defined as systems composed of interacting agents that are diverse in both form and capabilities [5]. The individual agent's behavior is governed by a collection of rules. These agents respond to stimuli and the stimulus-response exhibited is a result of the rules embedded in the agent [5].

The agents can be seen as interactive systems communicating between themselves and the environment. All interactions are between the agents and between the agents and the environment. These interactions cannot be anticipated as the modifications induced in the environment by the agents' actually influence the future interactions of these agents. Interactive systems due to their undeterministic nature can never be fully specified nor fully tested. Business systems that have been developed that incorporate interactive software agents can be categorized as complex adaptive systems. CAS are defined by Holland as a system composed of interacting agents that respond to stimuli and are characterised by stimulus-response behavior that can be defined in terms of rules [2],[3],[5].



Major improvements in adaptivity and robustness can be achieved in software solutions by integrating the typical properties and mechanisms of CAS defined by Holland [2],[3],[5]. Business systems are ideal for implementation as a CAS as they have severely constrained business rule sets in comparison to other CAS systems such as ant colonies, immune systems, cities and economies [2],[3].

The model proposed in this paper takes advantage of the idea that the undeterministic interactions between agents and between agents and their environment may result in emergent behavior a fundamental feature of a CAS. The model also utilises the idea of creating dynamic rules for the agents that allows them to respond to certain stimuli in the environment. The response to certain stimuli in the environment allows the agents to react with each other and with their environment.

## 5 ACEUS Agent Layers

Just as the ants or bees are the individual agents in their colonies and hives, the organic and insilica agents are the individual agents in this framework. Thus the ACEUS model has two distinct agent layers, the organic agent layer and the insilica agent layer as seen in figure 1.

The organic agent layer houses the organic and interfacing agents. Organic agents in ACEUS are simply the human users of the system. The ACEUS model sees all human interaction in the model as interactions of agents in the environment. The model does not see the difference between an organic agent and an insilica agent. Organic agents and interfacing agents are the main players in the organic agent layer. Organic agents are able to request certain tasks or make recommendations in ACEUS by depositing pheromone messages in the stigmergy layer via the interfacing agent. All interaction or communication from the organic agents is via the interfacing agents to the stigmergy layer. For example an organic agent will have to place a pheromone in the stigmergy layer via the interfacing agent to communicate with the inner layers. The limit of the organic agents is that the organic agent can only interact with the collective via an interfacing agent. Thus the organic agent is limit to the functionality that the interfacing agent can offer.

Interfacing agents are organic to insilica interfacing agents. These agents are responsible for depositing either pheromones or objects into the stigmergy layer. Interfacing agents can be create as PHP or JSP web applications that interact with a web server in order to communicate with the stigmergy layer. They can also be integrated into Java, C# or C++ application that the organic agents can use to communicate with the stigmergy layer as seen in figure 1. The organic agent's only link to the system is via the interfacing agents. For example if an organic agent wants to initiate a request or recommendation in the system, the organic agent will use the interfacing agent to set that request up. The interfacing agent will then drop the request in the form of a signaling object into the stigmergy layer. The interfacing agent will contain all the traditional graphical user interfaces (GUI). GUIs are needed for the organic agent to interact with the interfacing agent. All data about the environment needed to create pheromones or objects will be given to the organic agent via the interfacing agent. Interfacing agents can vary from full-blown applications to small HTML pages.

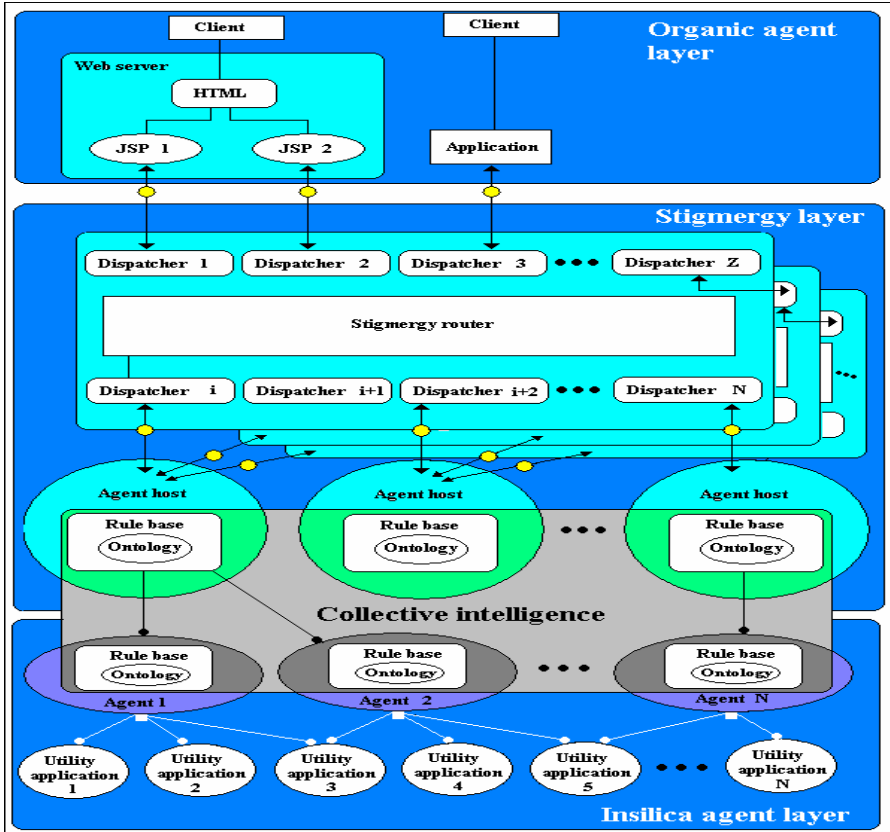


Fig. 1. The stigmergy layer model inside the ACEUS framework

All the insilica agents in the ACEUS model are housed in the insilica layer. These insilica agents are responsible for the key tasks in the environment being simulated. These agents communicate indirectly with each other via the stigmergy layer. Communication takes place by depositing pheromones or objects in the stigmergy layer. Communication between the agents is vital in this model even though the agents are unaware of one another and the communication is indirect. In the ACEUS model detecting different pheromones stimulates the agents. The response of the agent to the stimuli is governed by the rules built into the agent. Every insilica agent in the framework inherits from a template agent. The template agent contains all functionality that is common among all the agents. The template agent will basically contain an ontology and a rules engine that utilises a RETE algorithm [10]. When an agent is activated in ACEUS the ontology file along with the rules file for the agent is loaded into the agent's rules engine. The reason for this is the rules are dependent on the concepts and entities defined in the ontology files. The rules engine will be used to assert and retract facts as well as fire the agent's rules. The rules engine acts as the agent's brain. The agent's ability to respond to certain pheromone stimuli is due to its built in ontology and rule set.

The simple intelligent behavior displayed by the agent is a result of rules firing in the rules engine. The intelligence provided by the rules engine also allows for the communication mechanism i.e. to deposit and detect pheromones or objects in the stigmergy layer. Individual insilica agents will have their own set of rules that will be based on the concepts in the agent's ontology. If a new agent is to be added to the framework, the designer simply has to inherit from the template agent and design a new ontology and rule set for the agent. An onboard rules engine is used instead of global rules engine so that the agents can act as independent executing entities that may be distributed on different machines all over a computer network.

No disassembling or redesign is needed in a system based on the ACEUS model when the system experiences change. In order to incorporate new business functionality into the system the business logic is created in an ontology and rules set and added to a new agent. This agent is then deployed into the insilica layer of ACEUS. This robustness and flexibility in ACEUS is due to the agents in the framework being unaware of each other. The only interaction or communication between the agents is via the stigmergy layer. If this framework was implemented in an organization and the organization decided to extend the domain of the framework to incorporate new entities the system would be adaptive and robust enough to handle such a transition without any disassembling or redesign.

## 6 Agent's Rules and Ontology

The rules engine utilized for the agents was JESS (Java Expert System Shell) [11]. All the rules developed for each agent were placed in a rules file which was called the rule set. The rule set could be edited i.e. new intelligence could be added to the agent dynamically without effecting the overall system. It should be mention here that the designer of the system utilises the framework from the collective's perspective. The designer of the system is not an organic agent in the system and does not interact in the system once it is active. The designer of the system is able to comprehend the collective's intelligence where as any agent interacting in the system is not. When an agent is instantiated or activated the agent will firstly load up its rule set into its rule engine. All the different agent rule sets build up the rule base of the collective. The rule base also allows the designers easy access to the rules for the collective. A designer does not have to delve into the code of each agent to change its intelligence as in present systems. The designer can simply access the rule base where the entire collective's intelligence is accessible. Rules form a powerful means for developing intelligence into different agents. The general rule sets can also be defined in the rule base. General rule sets are rules that are used by a number of agents i.e. this is shared intelligence.

The ontology is the agent's vocabulary. Every entity and object known to the agent in the domain will be defined in the agent's ontology. Similar to the rule base the ontology base is the ontology for the entire collective. The ontology base is built up from all the agent's ontologies. What is interesting about the ontology base and rule base is due to the distribution of the segments of both these bases into to the agents no agent will ever be able to comprehend the collective's intelligence or have any understanding of the complete rule base or ontology base. At most an individual agent will

only be able to comprehend a small segment of the rule base or ontology base. The individual agents will never understand the collective as a whole they would only comprehend their main chores and abilities.

## 7 Modeling Stigmergy

When trying to model the stigmergy mechanism it is important to note that small perturbations caused by the individuals in the collective may not have an impact on the collective at all. However, in a certain state of the environment i.e. if certain criteria in the environment have been met, a small perturbation made by an individual may result in the creation of an enormous change in the environment.

From a stigmergy modeling perspective the state of the environment may act as stimulus to the individuals existing in it. The individuals (i.e. agents) have a rule set that can be triggered by the environment. Thus changes in the environment trigger certain rules in the individuals. Similarly, the environment also needs a rule base that governs the laws in the environment. An individual may change properties in the environment that cause rules to fire in the environment's rule base. This change in the environment may also stimulate the firing of rules in other individuals in the collective.

As an example of this consider the natural creation of a termite nest by a colony of termites. The termites randomly carry around and drop small soil pellets in the environment. This activity can be seen as a small perturbation in the environment. Due to the proximity of the area in which the termites are interacting a reasonable assumption would be that eventually a termite might drop a soil pellet next to another soil pellet. There may have been several thousand small perturbations of dropping soil pellets in the environment that had no effect on the collective. However, this single small perturbation of placing a soil pellet next to another soil pellet in the environment allows the environment to trigger a collective response. The rule in the environment induces a lower potential at the location of the two soil pellets. This lower potential in the environment stimulates the individuals of the collective to respond by dropping their soil pellets at this point of lower potential. The idea of lower potentials being induced in the environment is similar to the ideas in co-fields model for swarm intelligence [12].

## 8 The Stigmergy Layer Model

The Stigmergy layer is the third layer in the ACEUS model. The complete stigmergy layer inside the ACEUS model can be seen in figure 1. The stigmergy layer consists of two major components, the stigmergy routers and agent hosts.

### 8.1 Stigmergy Router

In order to utilize the stigmergy layer, clients or organic agents (as referred to in the ACEUS system) will have to access the stigmergy layer via an interfacing agent i.e. an application or web interface as seen in figure 1. The former is a software application that utilizes the ACEUS API to access the stigmergy layer while the latter is a HTML page that utilizes java server pages (JSP) to access the ACEUS API. In the ACEUS API there is a class called stigmergy client class that allows remote machines

to access the stigmergy layer via TCP/IP. The stigmergy client class contacts the stigmergy layer and requests a stigmergy router. If the stigmergy client class specifies a specific stigmergy router the client is connected to that stigmergy router. If no stigmergy router is specified then the stigmergy router that will best serve the request is connected to the client.

A stigmergy layer router can be created on any machine by executing a stigmergy router server application on the machine. The stigmergy router server application is part of the ACEUS package. The stigmergy layer router consists of a number of dispatchers. Each dispatcher is assigned to a connecting client or an agent host. Once an organic agent has made a connection to the stigmergy router a unique dispatcher is created for the agent. Similarly, the stigmergy router creates a unique dispatcher for any agent host that is instantiated or any other stigmergy router that is created. Thus each stigmergy layer router has dispatchers for clients, agent hosts and other stigmergy layer routers that have connected to it as seen in figure 1.

To ensure complete decentralization in the model there is no centralized stigmergy layer router (see figure 2). When a stigmergy layer router is created a list of all the addresses of the existing stigmergy router's in the stigmergy layer is needed. This list can range from zero to several thousand. If the list is zero then the first stigmergy router is being created. The list of available stigmergy routers is hosted as a published service. From this list the stigmergy router will connect itself to some of the already existing stigmergy routes. In figure 2 (a) we see that stigmergy router A is connected

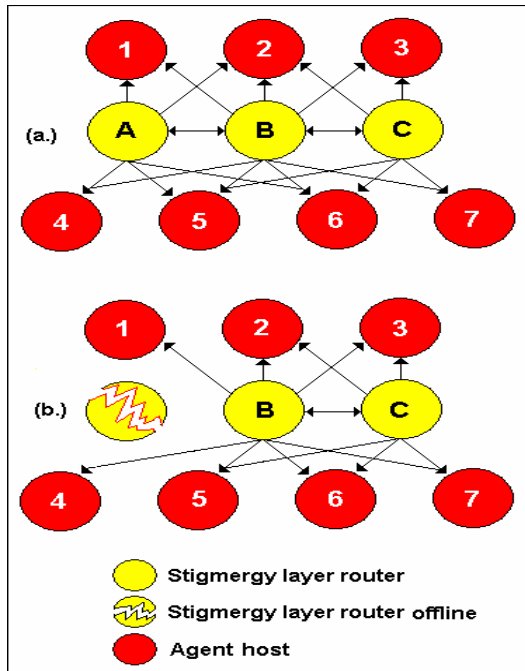


Fig. 2. Decentralization in a Stigmergy layer network

to stigmergy router B and stigmergy router B is connected to stigmergy router C. Similarly, when agent hosts are created on a machine the actual agent host also utilizes this list to gather information on the stigmergy routers that it may connect to. The agent host will then register on at least two of the stigmergy routers. In figure 2 (a) we see that agent host 1 has registered on stigmergy router A and B while agent 2 has registered with stigmergy router A, B and C. This architecture ensures that when one stigmergy router goes off line that the agent hosts are still accessibility via another stigmergy router as seen in figure 2 (b).

## 8.2 Agent Host

Agent hosts are individual machines that act as hosts to several agents. Agent hosts share part of their processing time and memory with the collective. An agent host has an onboard intelligence engine that allows the agent host to utilize an ontology and rule base that is embedded into the agent host. The ontology and rule base enables intelligence in the agent host that allows it the ability to route certain tasks to agents capable of processing the task. The ontology is an explicit conceptual specification of concepts and relationships known to the agent host [13],[14]. The rule base is a rule set built on the concepts and relationships in the ontology. In figure 1 the rule base is drawn encompassing the ontology, the reason for this is that the rule base cannot exist without the core ontology.

Each agent that is created on an agent host also has an independent onboard intelligence engine comprised of an ontology and a rule base (as discussed in section 5). Both the ontologies and rule bases of the agent hosts and agents are completely decentralized. The collective intelligence of the system is made up by the aggregation of all the ontologies and rule bases as seen in figure 1. The intelligence in the agent is utilized to facilitate a certain task where as the intelligence in the agent host facilitates workflow of a number of tasks and finding the correct agent for each task.

## 9 Messaging in the Stigmergy Layer

Messaging in the stigmergy layer happens via an artificial pheromone that is passed though the system via TCP/IP. The artificial pheromone is nothing more than a simple object containing data embedded in a XML file. The artificial pheromone can be deposited and detected in the system.

In natural systems where stigmergy is utilized the agents are directed by the activities taking place in their environment. In order to model such a mechanism in an artificial environment the activities taking place in the system would have to influence the behavior of the agents. In order for the agents to detect the state variation in the activities, the activities would have to influence the sensory inputs of the agents that in turn would respond to the stimulus. This stimulus response in the agents would be constructed by means of rules written in the agent's rule base. Rules governing the workflow of the activities would have to be written in the agent host's rule base, as the agent host is the artificial equivalent of the environment in a natural system

To construct such a mechanism the agent host would have to have the ability to inform the agents about the state of the environment i.e. itself. The agent host can utilise its intelligent system (i.e. rules engine and ontology) to deposit messages via a

pheromone into the agent's intelligent system. The agent then has rules that allow it to respond to the pheromones deposited in its intelligent system. Similarly the agent can also deposit pheromones in the agent host's intelligent system.

An agent depositing a pheromone in the agent host's intelligence system can initiate an entire workflow process. The initiated workflow process may then in turn regulate and control many other agents by means of the agent host depositing pheromones in their intelligent systems. Thus the actual workflow process is actually regulating the agents in the system similar to stigmergy mechanisms natural systems.

## 10 Conclusions

In this paper model called ACEUS has been presented that can be used in a business environment to build a software system that imitates a complex adaptive system (CAS) as well as synthesizes stigmergy for interactions between agents and agents and the environment. The design of ACEUS entailed three layers, namely: the organic agent layer, the insilica agent layer and the stigmergy layer. The layers allowed the model to be easily changed without disassembling or redesign of the entire system. The model's design allowed easy adaptability in terms of adding new agents as well as an easy way to access the systems ontology and rule base without accessing the individual agent's internal code.

The ACEUS model was created for environments that are growing at a rapid pace where the present methods of software design cannot keep up to the constant change. The present design structures in these environments are not robust or adaptable enough to cope with the constant change and expansion. The ACEUS model allows reuse of code, agents, concepts and rules as well as dynamic addition of these entities into the model. Designers can add entities to the model without having to have a complete understanding of all the entities in the model. The ACEUS framework is robust enough to be distributed over many machines or operate on a single machine. ACEUS allows thin client, thick client or web clients. For example interfacing agents can be designed to be simple HTML pages or full-blown applications.

One of the most promising ideas the ACEUS presents is that complex behavior patterns can emerge from individuals that are following simple rules. These simple rules are embedded into the individual agents. As the ACEUS model grows i.e. more and more agents are added the complexity of the system can increase dramatically. To avoid this problem the complexity of the rule base and ontology is hidden from the individual agents. Individual agents can only view a small segment of the rule base and a small segment of the ontology. The segment the individual agent can comprehend is the segment that is actually embedded into the individual agent. Thus a designer adding a new agent to the framework will only be concerned with the rules of the agent. The agent designer does not need to be concerned with all the other rules in the collective. The collective's ontology and rule base is abstract as a whole, as they are an aggregate of many smaller segments. The complete ontology and rule base can only be viewed from the collective's perspective. By incorporating a collective's and an individual's perspective reduces complexity in the ACEUS model e.g. viewing the system from a collective's perspective could be overwhelming in complexity whereas the view point from the individuals perspective would be very simple and easy to understand.

The model has been implemented in a mobile telecommunication environment to optimize the frequency assignment problem (FAP) [15]. In this scenario the implementation of ACEUS and in particular the use of stigmergy allowed the continuous improvement of the overall frequency interference in shorter time periods. The model was compared to the simulated annealing approach for solving this type of problem and in each experiment the simulated annealing approach was out performed both in quality and process time [15]. The ACEUS model has produced promising results when optimizing FAP [15]. The model showed enhancement in network quality and improved times when producing frequency plans.

## 11 Future of the Stigmergy Layer Model

The stigmergy layer model forms the bases of the ACEUS system. This model is in its infancy and may still under go major changes and rethinks, however there exists the potential in the model for simulating swarm intelligence and building the semantic web. The ultimate goal for the stigmergy layer model would be to lay the foundation for developing resilient and adaptive software for the foundation of the semantic web. The ACEUS framework has the potential of facilitating self-organization, a sort after characteristic in modern software systems and architectures. The first steps towards these goals have been instantiated. Future investigations on the stigmergy layer model and ACEUS framework will entail the creation of peer-to-peer (P2P) systems and a novel semantic web architecture on certain company intranets for example the configuration and optimization of a mobile telecommunications network.

## References

1. Bonabeau, E. and Meyer, C., “*Swarm Intelligence: A Whole New Way to Think About Business*”, Harvard Business Review, Harvard Business School Publishing Corporation, (2001).
2. Sutherland, J. “*Business Object and Component Architectures: Enterprise Application Integration Encounters Complex Adaptive Systems*”, HICSS-34 Outrigger Wailea Resort Maui, January, (2001).
3. Sutherland, J. and van den Heuvel W., “*Enterprise Application Integration Encounters Complex Adaptive Systems: A Business Object Perspective*”, 35<sup>th</sup> Annual Hawaii International Conference on System Sciences (HICSS’02) Big Island, Hawaii, Volume 9, January, (2002).
4. Bonabeau, E., Dorigo, M. and Theraulaz, G., “*Swarm Intelligence From Natural to Artificial Systems*”, Santa Fe Institute, Studies in the science of complexity, Oxford University Press, New York, (1999).
5. Holland, J.H., “*Hidden Order: How Adaption Builds Complexity*”, Helix Books, 1995.
6. Montresor, A., Meling, H., and Babaoğlu, Ö., “*Towards Adaptive, Resilient and Self-Organizing Peer-to-Perr Systems*”, Technical Report UBLCS-2002-09, Dept. of Computer Science, University of Bologna, September, (2002).
7. Montresor, A., Meling, H., and Babaoğlu, Ö., “*Toward Self-Organizing, Self-Repairing and Resilient Large-Scale Distributed Systems*”, Technical Report UBLCS-2002-10, Dept. of Computer Science, University of Bologna, September, (2002).



8. Van Dyke Parunak, H., Brueckner, S. and Sauter, J. “*Digital pheromone mechanisms for coordination of unmanned vehicles*”. In 1st International Joint Conference on Autonomous Agents and Multiagent Systems AAMAS’02, ACM Press, (2002), 449–450.
9. Valckenaers, P., Kollingbaum, M., Van Brussel, H., Bochmann O. and Zamfirescu C., “The Design of Multi-Agent Coordination and Control Systems using Stigmergy”, Proc. of the IWES’01 Conference, (2001). <http://www.csd.abdn.ac.uk/~mkolling/publications/DesignMultiAgentCoordControlStigmergy.pdf>
10. Forgy C. L., “Rete: A Fast Algorithm for the Many pattern/Many Object-Pattern Matching Problem”, Artificial Intelligence, Vol. 19, no. 1, September, (1982), 17-37.
11. JESS, The Java Expert System Shell, E.J. Friedman-Hill, Distributed Computing Systems, Sandia National Laboratories, Verion 5.0, January (2000).
12. <http://herzberg.ca.sandia.gov/jess/>
13. Leonardi, L., Mamei, M., and Zambonelli, F., “*Co-Fields: Towards a Unified Model for Swarm Intelligence*” (2002). <http://polaris.ing.unimo.it/Zambonelli/Swarm.pdf>
14. Gruninger M. and Lee J., “*Ontology Applications and Design*”, Communication of the ACM, Vol. 45, No.2, February, (2002), 39-41.
15. Mizoguchi, R., “*Ontology-Based Systematization of Functional Knowledge*”, Proceeding of the TMCE, Wuhan, China, April, (2002), 45-64.
16. O’Reilly G.B. and Ehlers E.M., “*Utilizing the Swarm Effect to Optimize the Frequency Assignment Problem*”, Submitted for publication, January, (2006).

# Model Checking for Epistemic and Temporal Properties of Uncertain Agents<sup>\*</sup>

Zining Cao

Department of Computer Science and Engineering  
Nanjing University of Aero. & Astro., Nanjing 210016, China  
caozn@nuaa.edu.cn

**Abstract.** In this paper, we introduce a probabilistic epistemic temporal logic, called *PETL*, which is a combination of temporal logic and probabilistic knowledge logic. The model checking algorithm is given. Furthermore, we present a probabilistic epistemic temporal logic, called  $\mu$ *PETL*, which generalizes  $\mu$ -calculus by adding probabilistic knowledge modality. Similar to  $\mu$ -calculus,  $\mu$ *PETL* is a succinct and expressive language. It is showed that temporal modalities such as “always”, “some-time” and “until”, and probabilistic knowledge modalities such as “probabilistic knowledge” and “probabilistic common knowledge” can be expressed in such a logic. *PETL* is proven to be a sublogic of  $\mu$ *PETL*. The model checking technique for  $\mu$ *PETL* is also studied.

## 1 Introduction

The design of systems that are required to perform high-level management and control tasks in complex dynamic environments is becoming of increasing commercial importance. Such systems include the management and control of air traffic systems, telecommunications networks, business processes, space vehicles, and medical services. Experience in applying conventional software techniques to develop such systems has shown that they are very difficult and very expensive to build, verify, and maintain. Agent-oriented systems, based on a radically different view of computational entities, offer prospects for a qualitative change in this position. A number of different approaches have emerged as candidates for the study of agent-oriented systems. One main approach is concerned with the formal representation of the mental attitudes of autonomous entities, or agents, in a distributed system. For this task several modal logics have been developed in the past 20 years, the most studied being logics for knowledge, beliefs, desires, goals, and intentions. These logics are seen as specifications of particular classes of *MAS* systems. Their aim is to offer a description of the macroscopic mental properties (such as knowledge, belief and etc.) that a *MAS* should exhibit in a specific class of scenarios. A considerable number of these formal studies are available in the literature and temporal extensions of these (i.e., combinations

---

<sup>\*</sup> This work was supported by the National Science Foundation of China under Grant 60473036.

of *CTL* or *LTL* with the modalities for the mental attitudes) have appeared recently [6,13,17].

Verification of reaction systems by means of model checking techniques is now a well-established area of research [4]. In this paradigm one typically models a system  $S$  in terms of automata (or by a similar transition-based formalism), builds an implementation  $P_S$  of the system by means of a model-checker friendly language such as the input for *SMV* or *PROMELA*, and finally uses a model-checker such as *SMV* or *SPIN* to verify certain temporal property  $\varphi$  the system:  $M_P \models \varphi$ , where  $M_P$  is a temporal model representing the executions of  $P_S$ . As it is well known, there are intrinsic difficulties with the naive approach of performing this operation on an explicit representation of the states, and refinements of symbolic techniques (based on *OBDD*'s, and *SAT* translations) are being investigated to overcome these hurdles. Formal results and corresponding applications now allow for the verification of complex systems that generate more than  $10^{20}$  states.

The field of multi-agent systems has also recently become interested in the problem of verifying complex systems. In *MAS*, modal logics representing concepts such as knowledge, belief, and intention. Since these modalities are given interpretations that are different from the ones of the standard temporal operators, it is not straightforward to apply existing model checking tools developed for *LTL*\ *CTL* temporal logic to the specification of *MAS*. The recent developments of model checking *MAS* can broadly be divided into streams: in the first category standard predicates are used to interpret the various intensional notions and these are paired with standard model checking techniques based on temporal logic. Following this line is [22] and related papers. In the other category we can place techniques that make a genuine attempt at extending the model checking techniques by adding other operators. Works along these lines include [2,14] and so on. Given that agents work in unknown environments, and interact with other agents that may, in turn, be unpredictable, then it is essential for any formal agent description to incorporate some mechanism for capturing this aspect. Within the framework of executable specifications, formal descriptions involving uncertainty must also be executable. To express the probabilistic epistemic property in *MAS*, Ferreira, Fisher and Hoek introduced probabilistic epistemic temporal logic *PROTEM* in [7,8], which is a combination of temporal logic and probabilistic belief logic *P<sub>F</sub>KD45* [7]. For example, one can express statements such as “if it is probabilistic common knowledge in group of agents  $\Gamma$  that  $\varphi$ , then  $\Gamma$  can achieve a state satisfying  $\psi$ ”. Kooi’s work [18] combined the probabilistic epistemic logic with the dynamic logic yielding a new logic, *PDEL*, that deals with changing probabilities and takes higher-order information into account. The syntax of *PDEL* is an expansion of probabilistic epistemic logic by introducing dynamic logic formulas. The semantics of *PDEL* is based on a combination of Kripke structure and probability functions.

In this paper, we first introduce a probabilistic epistemic temporal logic *PETL* which is similar to *P<sub>F</sub>KD45* but extend it with probabilistic common knowledge. Then we present a probabilistic epistemic temporal logic  $\mu$ *PETL*, which is an

extension of  $\mu$ -calculus by adding probabilistic knowledge modality  $K_a^p$ . Although its syntax is very simple, we can show that temporal modalities such as “always”, “sometime” and “until”, and probabilistic knowledge modalities such as “everyone knows with probability” and “probabilistic common knowledge” can be expressed in such a logic. In fact we prove that *PETL* is a sublogic of  $\mu$ *PETL*. A translating function from any *PETL* formula to an equivalent  $\mu$ *PETL* formula is given. For temporal logics, it is well known that almost all famous temporal logics, such as *PDL*, *LTL*, *CTL* and *CTL\**, are sublogic of  $\mu$ -calculus, hence the problems of model checking *PDL*, *LTL*, *CTL* and *CTL\** can be uniformly reduced to model checking  $\mu$ -calculus [3]. Model checking  $\mu$ -calculus is a very active research area and there have been lots of algorithm for model checking  $\mu$ -calculus [3]. Similarly, for temporal epistemic logics,  $\mu$ *PETL* plays on a role of  $\mu$ -calculus. Almost all model checking problems for probabilistic epistemic temporal logics such as *PETL*, *PROTEM* [7,8] and *PDEL* [18] can be reduced to model checking  $\mu$ *PETL*. Hence it is important to study model checking algorithms for  $\mu$ *PETL*. In fact, it is not difficult to extend  $\mu$ -calculus model checking algorithm to  $\mu$ *PETL*. In this paper, we present a model checking algorithm for  $\mu$ *PETL*. We also study the complexity of model checking  $\mu$ *PETL*.

The rest of the paper is organized as follows: In Section 2, we present a probabilistic epistemic temporal logic *PETL*, give its syntax, semantics and model checking algorithm. In Section 3, we propose a probabilistic epistemic temporal logic  $\mu$ *PETL*. *PETL* is showed to be a sublogic of  $\mu$ *PETL*. The approach to model checking  $\mu$ *PETL* is studied. Furthermore, we study the complexity of model checking  $\mu$ *PETL*. The paper is concluded in Section 4.

## 2 Probabilistic Epistemic Temporal Logic *PETL*

In this section, we introduce a probabilistic epistemic temporal logic *PETL* which can be viewed as a variant of *PROTEM* [8]. The basic formulas of *PROTEM* can be classified into two categories: the standard belief logic formula such as  $B_a\varphi$ , and the probability formulas such as  $P_x^\geq$ ,  $P_{x+y}^\geq$  and etc. The formula  $B_a^p(\varphi)$  is an abbreviation for  $B_a(P_p^\geq(\varphi))$ , intuitively, this says that “agent  $a$  believes that the probability of  $\varphi$  is greater than or equal to  $p$ ”. A main shortcoming of *PROTEM* is that probabilistic common knowledge is not part of the system. In *PETL*,  $K_a^p\varphi$  is a basic formula, which makes the syntax and inference system simpler. Moreover, probabilistic common knowledge operator is included.

### 2.1 Syntax of *PETL*

The well form formulas of *PETL* are defined as follows.

**Definition 1.** The set of formulas in *PETL*, called  $L^{PETL}$ , is given by the following rules:

- (1) If  $\varphi \in$  atomic formulas set  $\Pi$ , then  $\varphi \in L^{PETL}$ .
- (2) If  $\varphi \in$  proposition variables set  $V$ , then  $\varphi \in L^{PETL}$ .

- (3) If  $\varphi \in L^{PETL}$ , then  $\neg\varphi \in L^{PETL}$ .
- (4) If  $\varphi, \psi \in L^{PETL}$ , then  $\varphi \wedge \psi \in L^{PETL}$ .
- (5) If  $\varphi, \psi \in L^{PETL}$ , then  $\bigcirc\varphi, \square\varphi, \varphi U\psi \in L^{PETL}$ . Intuitively,  $\bigcirc$  means next,  $\square$  means always and  $U$  means until.
- (6) If  $\varphi \in L^{PETL}$ , then  $K_a^p\varphi, E_\Gamma^p\varphi, C_\Gamma^p\varphi \in L^{PETL}$ , where  $a \in Agent, \Gamma \subseteq \Sigma, p \in [0, 1]$ . Intuitively,  $K_a^p\varphi$  means that agent  $a$  knows the probability of  $\varphi$  is no less than  $p$ .  $E_\Gamma^p\varphi$  means that every agent in  $\Gamma$  knows the probability of  $\varphi$  is no less than  $p$ .  $C_\Gamma^p\varphi$  means that “the probability of  $\varphi$  is no less than  $p$ ” is a common knowledge by every agent in  $\Gamma$ .

The following abbreviations are used:

$K_a^{>p}\varphi \stackrel{def}{=} \neg K_a^{1-p}\neg\varphi$ , here  $K_a^{>p}\varphi$  means that agent  $a$  knows the probability of  $\varphi$  is greater than  $p$ .

$K_a^{<p}\varphi \stackrel{def}{=} \neg K_a^p\varphi$ , here  $K_a^{<p}\varphi$  means that agent  $a$  knows the probability of  $\varphi$  is less than  $p$ .

$K_a^{\leq p}\varphi \stackrel{def}{=} K_a^{1-p}\neg\varphi$ , here  $K_a^{\leq p}\varphi$  means that agent  $a$  knows the probability of  $\varphi$  is no more than  $p$ .

$K_a^{=p}\varphi \stackrel{def}{=} K_a^p\varphi \wedge K_a^{1-p}\neg\varphi$ , here  $K_a^{=p}\varphi$  means that agent  $a$  knows the probability of  $\varphi$  is equal to  $p$ .

Similarly, we can define  $E_\Gamma^{>p}\varphi, C_\Gamma^{=p}\varphi$ , and etc. Thus using  $K_a^p\varphi, E_\Gamma^p\varphi$  and  $C_\Gamma^p\varphi$ , we can express various of probabilistic knowledge properties.

## 2.2 Semantics of *PETL*

We will describe the semantics of *PETL*, that is, a formal model that we can use to determine whether a given formula is true or false.

Any transition-based semantics allows for the representation of temporal flows of time by means of a successor relation. For example, *CTL* is interpreted on plain Kripke models. To work with a probabilistic epistemic temporal language, we need to consider a semantics that also allows for the automatic representation of the epistemic relations between computational states [10]. The mainstream semantics that allows one to do so is the one of interpreted systems [10].

**Definition 2.** Given a set of agents  $A = \{1, \dots, n\}$ , a probabilistic epistemic temporal model (or simply a model) is a tuple  $S = (Q, T, \sim_1, \dots, \sim_n, P_1, \dots, P_n, V)$ , where  $Q$  is the set of the global states for the system (henceforth called simply states);

$T \subseteq Q \times Q$  is a total binary (successor) relation on  $G$ ;

$\sim_a \subseteq Q \times Q$  ( $a \in A$ ) is an epistemic accessibility relation for each agent  $a \in A$  defined by  $s \sim_a s'$  iff  $l_a(s) = l_a(s')$ , where the function  $l_a: Q \rightarrow L_a$  returns the local state of agent  $a$  from a global state  $s$ ; obviously  $\sim_a$  is an equivalence relation;

$P_a$  is a probability function:  $Q \times \wp(Q) \rightarrow [0, 1]$ , such that for every  $a, P_a(s, \{s' | l_a(s) = l_a(s')\}) = 1$ .

$V : Q \rightarrow 2^{PV_K}$  is a valuation function for a set of propositional variables  $PV_K$  such that  $true \in V(s)$  for all  $s \in Q$ .  $V$  assigns to each state a set of propositional variables that are assumed to be true at that state.

We can now turn to the definition of semantics of *PETL*.

*Computations.* A computation in  $M$  is a possibly infinite sequence of states  $\pi = (s_0, s_1, \dots)$  such that  $(s_i, s_{i+1}) \in T$  for each  $i \in N$ . Specifically, we assume that  $(s_i, s_{i+1}) \in T$  iff  $s_{i+1} = t(s_i, act_i)$ , i.e.,  $s_{i+1}$  is the result of applying the transition function  $t$  to the global state  $s_i$ , and an action  $act_i$ . In the following we abstract from the transition function, the actions, and the protocols, and simply use  $T$ , but it should be clear that this is uniquely determined by the interpreted system under consideration. For a computation  $\pi = (s_0, s_1, \dots)$ , let  $\pi[k] = s_k$ , and  $\pi_k = (s_0, \dots, s_k)$ , for each  $k \in N$ . By  $\Pi(s)$  we denote the set of all the infinite computations starting at  $s$  in  $M$ .

Formally, a formula  $\varphi$  is interpreted as a set of states in which  $\varphi$  is true. We write such set of states as  $[[\varphi]]_S$ , where  $S$  is a model. The set  $[[\varphi]]_S$  is defined recursively as follows:

**Definition 3.** Semantics of *PETL*

$$\begin{aligned}
[[p]]_S &= \{q \mid p \in \pi(q)\}; \\
[[\neg\varphi]]_S &= Q - [[\varphi]]_S; \\
[[\varphi \wedge \psi]]_S &= [[\varphi]]_S \cap [[\psi]]_S; \\
[[\bigcirc\varphi]]_S &= \{q \mid \text{for all computations } \pi \in \Pi(q), \text{ we have } \pi[1] \in [[\varphi]]_S.\}; \\
[[\square\varphi]]_S &= \{q \mid \text{for all computations } \pi \in \Pi(q) \text{ and all positions } m \geq 0, \text{ we have } \pi[m] \in [[\varphi]]_S.\}; \\
[[\varphi U \psi]]_S &= \{q \mid \text{for all computations } \pi \in \Pi(q), \text{ there exists a position } m \geq 0, \\
&\text{ such that } \pi[m] \in [[\psi]]_S \text{ and for all positions } 0 \leq j < m, \text{ we have } \lambda[j] \in [[\varphi]]_S.\}; \\
[[K_a^p\varphi]]_S &= \{q \mid P_a(q, \sim_a(q) \cap [[\varphi]]_S) \geq p\}, \text{ here } \sim_a(q) = \{r \mid (q, r) \in \sim_a\}; \\
[[E_T^p\varphi]]_S &= \bigcap_{a \in T} [[K_a^p\varphi]]_S; \\
[[C_T^p\varphi]]_S &= \bigcap_{k \geq 1} [[(F_T^p)^k\varphi]]_S, \text{ here } [[(F_T^p)^0\varphi]]_S = Q, [[(F_T^p)^{k+1}\varphi]]_S = [[E_T^p(\varphi \wedge \\
&(F_T^p)^k\varphi)]_S.
\end{aligned}$$

### 2.3 Model Checking for *PETL*

In [8], a probabilistic epistemic temporal logic *PROTEM* was presented, but the model checking algorithm are not studied. In this section we give a model checking algorithm for *PETL*. The model checking problem for *PETL* asks, given a model  $S$  and a *PETL* formula  $\varphi$ , for the set of states in  $Q$  that satisfy  $\varphi$ . In the following, we denote the desired set of states by  $Eval(\varphi)$ .

For each  $\varphi'$  in  $Sub(\varphi)$  do

$$\begin{aligned}
\text{case } \varphi' = p &: Eval(\varphi') := Reg(p) \\
\text{case } \varphi' = \neg\theta &: Eval(\varphi') := Eval(true) - Eval(\theta) \\
\text{case } \varphi' = \theta_1 \wedge \theta_2 &: Eval(\varphi') := Eval(\theta_1) \cap Eval(\theta_2) \\
\text{case } \varphi' = \bigcirc\theta &: Eval(\varphi') := Pre(Eval(\theta))
\end{aligned}$$

```

case  $\varphi' = \llbracket \theta \rrbracket$  :
   $Eval(\varphi') := Eval(true)$ 
   $\rho_1 := \rho_2 := Eval(\theta)$ 
  repeat
     $Eval(\varphi') := Eval(\varphi') \cap \rho_1$ 
     $\rho_1 := Pre(Eval(\varphi')) \cap \rho_2$ 
  until  $\rho_1 = Eval(\varphi')$ 
case  $\varphi' = \theta_1 U \theta_2$  :
   $Eval(\varphi') := Eval(false)$ 
   $\rho_1 := Eval(\theta_1)$ 
   $\rho_2 := Eval(\theta_2)$ 
  repeat
     $Eval(\varphi') := Eval(\varphi') \cup \rho_2$ 
     $\rho_2 := Pre(Eval(\varphi')) \cap \rho_1$ 
  until  $\rho_1 = Eval(\varphi')$ 
end case
case  $\varphi' = K_a^p \theta$  :  $Eval(\varphi') := \{q \mid P_a(Img(q, \sim_a) \cap Eval(\theta)) \geq p\}$ 
case  $\varphi' = E_\Gamma^p \theta$  :  $Eval(\varphi') := \bigcap_{a \in \Gamma} Eval(K_a^p \theta)$ 
case  $\varphi' = C_\Gamma^p \theta$  :
   $Eval(\varphi') := Eval(true)$ 
  repeat
     $\rho := Eval(\varphi')$ 
     $Eval(\varphi') := \bigcap_{a \in \Gamma} (\{q \mid P_a(Img(q, \sim_a) \cap Eval(\theta) \cap \rho) \geq p\})$ 
  until  $\rho = Eval(\varphi')$ 
end case
return  $Eval(\varphi)$ 

```

The algorithm uses the following primitive operations:

- (1) The function *Sub*, when given a formula  $\varphi$ , returns a queue of syntactic subformulas of  $\varphi$  such that if  $\varphi_1$  is a subformula of  $\varphi$  and  $\varphi_2$  is a subformula of  $\varphi_1$ , then  $\varphi_2$  precedes  $\varphi_1$  in the queue *Sub*( $\varphi$ ).
- (2) The function *Reg*, when given a proposition  $p \in \Pi$ , returns the set of states in  $Q$  that satisfy  $p$ .
- (3) The function *Pre*, when given a set  $\rho \subseteq Q$  of states, returns the set of states  $q$  such that from  $q$  the next state to lie in  $\rho$ . Formally, *Pre*( $\rho$ ) contains state  $q \in Q$  such that  $(q, s) \in T$  where  $s \in \rho$ .
- (4) The function *Img* :  $Q \times 2^{Q \times Q} \rightarrow Q$ , which takes as input a state  $q$  and a binary relation  $R \subseteq Q \times Q$ , and returns the set of states that are accessible from  $q$  via  $R$ . That is,  $Img(q, R) = \{q' \mid qRq'\}$ .
- (5) Union, intersection, difference, and inclusion test for state sets. Note also that we write  $Eval(true, e)$  for the set  $Q$  of all states, and write  $Eval(false, e)$  for the empty set of states.

Partial correctness of the algorithm can be proved induction on the structure of the input formula  $\varphi$ . Termination is guaranteed since the state space  $Q$  is finite.

**Proposition 1.** The algorithm given in the above terminates and is correct, i.e., it returns the set of states in which the input formula is satisfied.

### 3 Probabilistic Epistemic Temporal Logic $\mu PETL$

In this section, we propose a logic called  $\mu PETL$ , which is a combination of  $\mu$ -calculus and probabilistic knowledge logic. It is well known that  $LTL$  and  $CTL$  are sublogics of  $\mu$ -calculus [4], hence temporal epistemic logic  $PETL$ , which is a combination of  $CTL$  and probabilistic knowledge logic, is also a sublogic of  $\mu PETL$ . Similarly, since  $PROTEM$  in [8] is essentially  $CTL$  with probabilistic knowledge operators, this logic is a sublogic of  $\mu PETL$ .

#### 3.1 Syntax of $\mu PETL$

Throughout this paper, we let  $L^{\mu PETL}$  be a language which is just the set of formulas of interest to us. In the following, we use  $\Sigma$  to denote the set of agents.

**Definition 4.** The set of formulas in  $\mu PETL$ , called  $L^{\mu PETL}$ , is given by the following rules:

- (1) If  $\varphi \in$  atomic formulas set  $\Pi$ , then  $\varphi \in L^{\mu PETL}$ .
- (2) If  $\varphi \in$  proposition variables set  $V$ , then  $\varphi \in L^{\mu PETL}$ .
- (3) If  $\varphi \in L^{\mu PETL}$ , then  $\neg\varphi \in L^{\mu PETL}$ .
- (4) If  $\varphi_1, \varphi_2 \in L^{\mu PETL}$ , then  $\varphi_1 \wedge \varphi_2 \in L^{\mu PETL}$ .
- (5) If  $\varphi \in L^{\mu PETL}$ , then  $\bigcirc\varphi \in L^{\mu PETL}$ .
- (6) If  $\varphi \in L^{\mu PETL}$ , then  $K_a^p\varphi \in L^{\mu PETL}$ , where  $a \in Agent$ ,  $p \in [0, 1]$ .
- (7) If  $\varphi(X) \in L^{\mu PETL}$ , then  $\mu X.\varphi(X) \in L^{\mu PETL}$ , here  $X$  occurs positively in  $\varphi(X)$ , i.e., all free occurrences of  $X$  fall under an even number of negations..

The model of  $\mu PETL$  is the same as  $PETL$ . We write such set of states in which  $\varphi$  is true as  $[[\varphi]]_S^e$ , where  $S$  is a model and  $e: V \rightarrow 2^Q$  is an environment. We denote by  $e[X \leftarrow W]$  a new environment that is the same as  $e$  except that  $e[X \leftarrow W](X) = W$ . The set  $[[\varphi]]_S^e$  is defined recursively as follows:

**Definition 5.** Semantics of  $\mu PETL$

$$\begin{aligned}
[[p]]_S^e &= \{q \mid p \in \pi(q)\} \\
[[X]]_S^e &= e(X) \\
[[\neg\varphi]]_S^e &= Q - [[\varphi]]_S^e \\
[[\varphi \wedge \psi]]_S^e &= [[\varphi]]_S^e \cap [[\psi]]_S^e \\
[[\bigcirc\varphi]]_S^e &= \{q \mid \text{for all computations } \pi \in \Pi(q), \text{ we have } \pi[1] \in [[\varphi]]_S^e.\} \\
[[K_a^p\varphi]]_S^e &= \{q \mid P_a(q, \sim_a(q) \cap [[\varphi]]_S^e) \geq p\}, \text{ here } \sim_a(q) = \{r \mid (q, r) \in \sim_a\}; \\
[[\mu X.\varphi(X)]]_S^e &= \bigcap \{W \subseteq Q \mid [[\varphi(X)]]_S^{e[X \leftarrow W]} \subseteq W\}.
\end{aligned}$$

Let  $S = (G, T, \sim_1, \dots, \sim_n, P_1, \dots, P_n, V)$  be a model. Notice that the set  $2^G$  of all subsets of  $G$  forms a lattice under the set inclusion ordering. Each element  $G'$  of the lattice can also be thought of as a predicate on  $G$ , where the predicate is



viewed as being true for exactly the states in  $G'$ . The least element in the lattice is the empty set, which corresponds to the predicate false, and the greatest element in the lattice is the set  $G$ , which corresponds to true. A function  $\tau$  mapping  $2^G$  to  $2^G$  is called a predicate transformer. A set  $G' \subseteq G$  is a fixed point of a function  $G': 2^G \rightarrow 2^G$  if  $\tau(G') = G'$ . Whenever  $\tau$  is monotonic (i.e., when  $U \subseteq V$  implies  $\tau(U) \subseteq \tau(V)$ ),  $\tau$  has a least fixed point denoted by  $\mu Z.\tau(Z)$ , and a greatest fixed point, denoted by  $\nu Z.\tau(Z)$ . When  $\tau$  is monotonic and  $\cup$ -continuous (i.e., when  $W_1 \subseteq W_2 \subseteq \dots$  implies  $\tau(\cup_i W_i) = \cup_i \tau(W_i)$ ), then  $\mu Z.\tau(Z) = \cup_i \tau^i(\text{False})$ . When  $\tau$  is monotonic and  $\cap$ -continuous (i.e., when  $W_1 \supseteq W_2 \supseteq \dots$  implies  $\tau(\cap_i W_i) = \cap_i \tau(W_i)$ ), then  $\nu Z.\tau(Z) = \cap_i \tau^i(\text{True})$ .

In order to characterize the properties of probabilistic knowledge, temporal and  $\mu$ -operator, we will characterize the formulas that are always true. More formally, given a model  $S$ , we say that  $\varphi$  is valid in  $S$ , and write  $S \models \varphi$ , if  $q \in [[\varphi]]_S^e$  for every state  $q$  in  $Q$ , and we say that  $\varphi$  is satisfiable in  $S$ , and write  $S, q \models \varphi$ , if  $q \in [[\varphi]]_S^e$  for some  $q$  in  $Q$ . We say that  $\varphi$  is valid, and write  $\models_{\mu PETL} \varphi$ , if  $\varphi$  is valid in all models, and that  $\varphi$  is satisfiable if it is satisfiable in some model. We write  $\Gamma \models_{\mu PETL} \varphi$ , if  $\varphi$  is valid in all models in which  $\Gamma$  is satisfiable.

### 3.2 Inference System

Now we list a number of valid properties of probabilistic knowledge, temporal modality and  $\mu$  operator, which comprise the inference system of  $\mu PETL$ .

*All instances of propositional tautologies and rules.*

$$K1. K_a^0 \varphi.$$

$$K2. K_a^p \varphi \wedge K_a^q \psi \rightarrow K_a^{\max(p+q-1, 0)} (\varphi \wedge \psi).$$

$$K3. K_a^p \varphi \rightarrow K_a^1 K_a^p \varphi.$$

$$K4. \neg K_a^p \varphi \rightarrow K_a^1 \neg K_a^p \varphi.$$

$$K5. K_a^p \varphi \rightarrow K_a^q \varphi, \text{ where } 1 \geq p \geq q \geq 0.$$

$$K6. K_a^{p+q} (\varphi \vee \psi) \rightarrow (K_a^p \varphi \vee K_a^q \psi), \text{ where } 1 \geq p + q \geq 0.$$

$$KT1. \vdash \varphi \Rightarrow \vdash K_a^1 \varphi.$$

$$KT2. \vdash \varphi \rightarrow \psi \Rightarrow \vdash K_a^p \varphi \rightarrow K_a^p \psi.$$

$$KT3. \vdash \neg(\varphi \wedge \psi) \Rightarrow \vdash \neg(K_a^p \varphi \wedge K_a^q \psi) \text{ for any } p, q \in [0, 1] \text{ such that } p + q > 1.$$

$$KT4. \vdash \neg(\varphi \wedge \psi) \Rightarrow \vdash K_a^p \varphi \wedge K_a^q \psi \rightarrow K_a^{p+q} (\varphi \vee \psi), \text{ where } p + q \leq 1.$$

$$KT5. \Gamma \vdash K_a^{p_n} \varphi \text{ for all } n \in M \Rightarrow \Gamma \vdash K_a^p \varphi, \text{ where } M \text{ is an arbitrary index}$$

set,  $p = \sup_{n \in M} (\{p_n\})$ .

$$KT6. \Gamma \cup (\cup_{\varphi \in \Sigma} (\{K_a^p \varphi \mid 0 \leq p \leq r\} \cup \{\neg K_a^q \varphi \mid 1 \geq q > r\})) \vdash \psi \text{ for any } r \in [0, 1] \\ \Rightarrow \Gamma \vdash \psi, \text{ where } \Sigma \text{ is a given set of formulas.}$$

$$M1. \varphi(\mu X. \varphi(X)) \rightarrow \mu X. \varphi(X).$$

$$MT1. \vdash \varphi(\psi) \rightarrow \psi \Rightarrow \vdash \mu X. \varphi(X) \rightarrow \psi.$$

$$N1. (\bigcirc \varphi \wedge \bigcirc (\varphi \rightarrow \psi)) \rightarrow \bigcirc \psi.$$

$$N2. \bigcirc \varphi \rightarrow \neg \bigcirc \neg \varphi.$$

$$NT1. \vdash \varphi \Rightarrow \vdash \bigcirc \varphi.$$

In this inference system,  $K1$ - $K6$ ,  $KT1$ - $KT6$  characterize probabilistic knowledge modality.  $M1$ ,  $MT1$  characterize least fixpoint operator.  $N1$ ,  $N2$  and  $NT1$

characterize temporal modality. A proof is said to be from  $\Gamma$  to  $\varphi$  if the premise is  $\Gamma$  and the last formula is  $\varphi$  in the proof. We say  $\varphi$  is provable from  $\Gamma$  in  $\mu PETL$ , and write  $\Gamma \vdash_{\mu PETL} \varphi$ , if there is a proof from  $\Gamma$  to  $\varphi$  in  $\mu PETL$ .

It is not difficult to prove that all axioms and rules in the inference system hold in any model  $S$ . Therefore we have the soundness of the inference system:

**Proposition 2.** The inference system of  $\mu PETL$  is sound, i.e.,  $\Gamma \vdash_{\mu PETL} \varphi \Rightarrow \Gamma \models_{\mu PETL} \varphi$ .

It is well known that  $M1$ ,  $MT1$ ,  $N1$ ,  $N2$  and  $NT1$  comprise a complete inference system for  $\mu$ -calculus [21]. Furthermore, in [5], we proved that  $K1$ - $K6$ , and  $KT1$ - $KT6$  comprise a complete inference system for probabilistic epistemic logic. Using the similar proof technique, we get the completeness of  $\mu PETL$ .

**Proposition 3.** The inference system of  $\mu PETL$  is complete, i.e.,  $\Gamma \models_{\mu PETL} \varphi \Rightarrow \Gamma \vdash_{\mu PETL} \varphi$ .

### 3.3 Expressivity of $\mu PETL$

Now we will discuss the express power of  $\mu PETL$ . As far as we know, the most temporal epistemic logics in the literature are sublogics of  $\mu PETL$ . We demonstrate this by an example, i.e.,  $PETL$ , is a sublogic of  $\mu PETL$ . Finally, we give a function which can translates an  $PETL$  formula into an equivalent  $\mu PETL$  formula.

We have seen that the syntax of  $\mu PETL$  is very simple, it is just an extension of  $\mu$ -calculus with knowledge modality  $K_a^p$ . But its express power is strong. In this section, we will prove that temporal operators  $\Box$  and  $(\cdot U \cdot)$ , knowledge operators  $E_T^p$  and  $C_T^p$  all can be expressed in  $\mu PETL$ . This means that  $PETL$  is a sublogic of  $\mu PETL$ .

**Proposition 4.** [4]  $\Box \varphi \models \nu X.(\varphi \wedge \bigcirc X)$  and  $\nu X.(\varphi \wedge \bigcirc X) \models \Box \varphi$ , here  $\nu X.\varphi(X) \stackrel{def}{=} \neg \mu X.\neg \varphi(\neg X)$ .

**Proposition 5.** [4]  $\varphi_1 U \varphi_2 \models \mu X.(\varphi_2 \vee (\varphi_1 \wedge \bigcirc X))$  and  $\mu X.(\varphi_2 \vee (\varphi_1 \wedge \bigcirc X)) \models \varphi_1 U \varphi_2$ .

Define  $(F_T^p)^0 \varphi = true$  and  $(F_T^p)^{k+1} \varphi = E_T^p(\varphi \wedge (F_T^p)^k \varphi)$ .

Then we take  $(S, s) \models C_T^p \varphi$  iff  $(S, t) \models (F_T^p)^k \varphi$  for all  $k \geq 1$ .

In [9],  $C_T^p \varphi$  was proven to be the greatest fixpoint solution of the equation  $X \leftrightarrow E_T^p(\varphi \wedge X)$ . So we have the following proposition.

**Proposition 6.**  $C_T^p \varphi \models \nu X.(E_T^p(\varphi \wedge X))$  and  $\nu X.(E_T^p(\varphi \wedge X)) \models C_T^p \varphi$ .

By the above propositions, all modal operators of  $PETL$  can be expressed in  $\mu PETL$ .

Now we can give a translating function from  $PETL$  formula to  $\mu PETL$  formula:

**Definition 6.** The translating function  $T$  is defined inductively as follows:

$$T(p) = p \text{ for atomic proposition } p$$

$$T(\neg \varphi) = \neg T(\varphi)$$

$$\begin{aligned}
T(\varphi_1 \wedge \varphi_2) &= T(\varphi_1) \wedge T(\varphi_2) \\
T(\bigcirc\varphi) &= \bigcirc T(\varphi) \\
T(\Box\varphi) &= \nu X.(T(\varphi) \wedge \bigcirc X) \\
T(\varphi_1 U \varphi_2) &= \mu X.(T(\varphi_2) \vee (T(\varphi_1) \wedge \bigcirc X)) \\
T(K_a^p \varphi) &= K_a^p T(\varphi) \\
T(E_\Gamma^p \varphi) &= \wedge_{a \in \Gamma} K_a^p T(\varphi) \\
T(C_\Gamma^p \varphi) &= \nu X.E_\Gamma^p(T(\varphi) \wedge X)
\end{aligned}$$

The following proposition states the correctness of translating function  $T$ .

**Proposition 7.** For any  $\varphi \in PETL$ ,  $T(\varphi) \in \mu PETL$  and  $\models T(\varphi) \leftrightarrow \varphi$ .

*Proof :* By Propositions 4, 5 and 6.

Since the temporal part of  $\mu PETL$  is the same as  $\mu$ -calculus, the temporal part of  $PETL$  is the same as  $CTL$  and  $\mu$ -calculus is more expressive than  $CTL$ , we have that  $\mu PETL$  is more expressive than  $PETL$ . It is not difficult to see that other temporal epistemic logics in the literature are sublogics of  $\mu PETL$  too. It is well known that  $PDDL$ ,  $LTL$  and  $CTL$  are sublogics of  $\mu$ -calculus [4], hence  $PROTEM$  in [8], which is essentially  $CTL$  with probabilistic epistemic operator, is also a sublogic of  $\mu PETL$ . Similarly, since  $PDEL$  [18] is a combination of  $PDL$  and probabilistic epistemic logics, this logic can also be expressed in  $\mu PETL$ .

### 3.4 Model Checking for $\mu PETL$

There are lots of model checking algorithms for  $\mu$ -calculus [4]. The only difference between the classical  $\mu$ -calculus and  $\mu PETL$  is the  $K_a^p$  operator. Hence almost all algorithms for  $\mu$ -calculus model checking can be modified to handle  $\mu PETL$  by implementing the functions which compute  $K_a^p$  operator. This means that we have in fact lots of model checking algorithms for  $\mu PETL$ .

In the following, we give a model checking algorithm for  $\mu PETL$ . We denote the desired set of states by  $Eval(\varphi, e)$ , where  $e$  is an environment.

For each  $\varphi'$  in  $Sub(\varphi)$  do

```

case  $\varphi' = p$  :  $Eval(\varphi', e) := Reg(p)$ 
case  $\varphi' = X$  :  $Eval(\varphi', e) := e(X)$ 
case  $\varphi' = \neg\theta$  :  $Eval(\varphi', e) := Eval(true, e) - Eval(\theta, e)$ 
case  $\varphi' = \theta_1 \wedge \theta_2$  :  $Eval(\varphi', e) := Eval(\theta_1, e) \cap Eval(\theta_2, e)$ 
case  $\varphi' = \bigcirc\theta$  :  $Eval(\varphi', e) := Pre(Eval(\theta, e))$ 
case  $\varphi' = K_a^p\theta$  :  $Eval(\varphi', e) := \{q \mid P_a(Img(q, \sim_a) \cap Eval(\theta, e)) \geq p\}$ 
case  $\varphi' = \mu X.\theta(X)$  :
     $Eval(\varphi', e) := Eval(false, e)$ 
    repeat
         $\rho := Eval(\varphi', e)$ 
         $Eval(\varphi', e) := Eval(\theta(X), e[X \leftarrow Eval(\varphi', e)])$ 
    until  $\rho = Eval(\varphi', e)$ 
end case
return  $Eval(\varphi, e)$ 

```

Notice that for any given inputs, all of these functions may be easily computed in time polynomial in the size of the inputs and structure against which they are being computed. In the case of  $\varphi' = \mu X.\theta(X)$ ,  $Eval(\varphi', e)$  is computed by iterative evaluation:  $[[\mu X.\varphi(X)]]_S^e = \cup_i \tau^i(\emptyset)$ , where  $\tau(W) = [[\varphi(X)]]_S^{e[X \leftarrow W]}$ . In fact, the complexity of this algorithm mainly depends on the cost of computing fixpoint.

Partial correctness of the algorithm can be proved induction on the structure of the input formula  $\varphi$ . Termination is guaranteed, because the state space  $Q$  is finite. Therefore we have the following proposition:

**Proposition 8.** The algorithm given in the above terminates and is correct, i.e., it returns the set of states in which the input formula is satisfied.

The modal  $\mu$ -calculus model checking problem is well known to be in the complexity class  $NP \cap co - NP$  [4]. Moreover, since  $\mu$ -calculus model checking is equivalent via linear time reductions to the problem of deciding the winner in parity game, in [15] the modal  $\mu$ -calculus model checking problem was proven to be in  $UP \cap co - UP$ , where  $UP$  is unambiguous nondeterministic polynomial time class and obviously  $P \subseteq UP \cap co - UP \subseteq NP \cap co - NP$ . This is the best result for the complexity of  $\mu$ -calculus model checking up to now. Since model checking algorithms for  $\mu$ -calculus can be modified to handle  $\mu PETL$  by implementing the functions which compute  $K_a^p$  operator. From a computational point of view, the complexity of  $\mu PETL$  model checking is the same as  $\mu$ -calculus. Therefore we have the following proposition:

**Proposition 9.** The  $\mu PETL$  model checking problem is in  $UP \cap co - UP$ .

Up to now, the best model checking algorithm for  $\mu$ -calculus costs subexponential time in the worst case [16]. Since the complexity of  $\mu PETL$  model checking is the same as  $\mu$ -calculus, we also have a subexponential time model checking algorithm for  $\mu PETL$ .

$PETL$  can also be verified uniformly by  $\mu PETL$  model checking algorithm. For  $PETL$ , since temporal operators and probabilistic common knowledge operator can be expressed by  $\mu$ -operator, we can translate any  $PETL$  formula into a  $\mu PETL$  formula, then reduce  $PETL$  model checking to  $\mu PETL$  model checking. Moreover,  $PROTEM$  [8] and  $PDEL$  [18] can also be verified by  $\mu PETL$  model checking algorithm since  $CTL$  and  $PDL$  were showed to be a sublogic of  $\mu$ -calculus [4].

## 4 Conclusions

Recently, there has been growing interest in the logics for representing and reasoning temporal and epistemic properties in multi-agent systems [2,8,13,14,17,18,22]. In this paper, we present a probabilistic epistemic temporal logic  $\mu PETL$ , which is a succinct and powerful language for expressing complex properties of multi-agent system. Similar to  $\mu$ -calculus in temporal logics, some probabilistic epistemic temporal logics can be translated into  $\mu PETL$ . As an example,  $PETL$  is showed to be a sublogic of  $\mu PETL$ . Thus a model checking

algorithm for  $\mu PETL$  can uniformly check various of temporal and probabilistic epistemic properties. In this paper, the approach to model checking for  $\mu PETL$  is studied. A famous efficient model checking technique is symbolic model checking [19], which uses ordered binary-decision diagrams (*OBDDs*) to represent Kripke structures. It is interesting to develop a symbolic approach for  $\mu PETL$  where the main problem is to give an *OBDD* style representation of probabilistic Kripke structures. This will be our further work. In some systems, predictable response times are essential for correctness. Such systems are called real-time systems. It is interesting to study whether  $\mu PETL$  can be extended to express real-time properties in real-time multi-agent systems. Some epistemic temporal logics and model checking algorithms have been used to verify the correctness of protocol systems [13,14,17]. It is also worth applying such  $\mu PETL$  logic and its model checking algorithm to such an area.

## References

1. F. Bacchus. Representing and reasoning with probabilistic knowledge: a logical approach to probabilities. Cambridge, Mass. : MIT Press, 1990.
2. M. Bourahla and M. Benmohamed. Model Checking Multi-Agent Systems. In *Informatica 29*: 189-197, 2005.
3. J. Bradfield and C. Stirling. Modal Logics and mu-Calculi: An Introduction. In *Handbook of Process Algebra*, Chapter 4. Elsevier Science B.V. 2001.
4. E. M. Clarke, J. O. Grumberg, and D. A. Peled. Model checking. The MIT Press, 1999.
5. Zining Cao, Chunyi Shi. Probabilistic Belief Logic and Its Probabilistic Aumann Semantics. *J. Comput. Sci. Technol.* 18(5): 571-579, 2003.
6. H. van Ditmarsch, W van der Hoek, and B. P. Kooi. Dynamic Epistemic Logic with Assignment, in *AAMAS-05*, 141-148, 2005.
7. N. de C. Ferreira, M. Fisher, W. van der Hoek: Practical Reasoning for Uncertain Agents. *Proc. JELIA-04, LNAI 3229*, 82-94, 2004.
8. N. de C. Ferreira, M. Fisher, W. van der Hoek: Logical Implementation of Uncertain Agents. *Proc. EPIA-05, LNAI 3808*, 536-547, 2005.
9. R. Fagin and J. Y. Halpern. Reasoning about knowledge and probability. *J. ACM*, 1994, 41(2): 340-367.
10. R. Fagin, J. Y. Halpern, Y. Moses and M. Y. Vardi. Reasoning about knowledge. Cambridge, Massachusetts: The MIT Press, 1995.
11. J. Y. Halpern and Y. Moses. Knowledge and common knowledge in a distributed environment. *J. ACM*, 1990, 37(3): 549-587.
12. W. van der Hoek. Some considerations on the logic PFD: A logic combining modality and probability. *J. Applied Non-Classical Logics*, 7(3): 287-307, 1997.
13. W. van der Hoek and M. Wooldridge. Model Checking Knowledge, and Time. In *Proceedings of SPIN 2002, LNCS 2318*, 95-111, 2002.
14. W. van der Hoek and M. Wooldridge. Cooperation, Knowledge, and Time: Alternating-time Temporal Epistemic Logic and its Applications. *Studia Logica*, 75: 125-157, 2003.
15. M. Jurdzinski. Deciding the winner in parity games is in  $UP \cap co-UP$ . *Information Processing Letters*, 68: 119-134, 1998.

16. M. Jurdzinski, M. Paterson and U. Zwick. A Deterministic Subexponential Algorithm for Solving Parity Games. In Proceedings of ACM-SIAM Symposium on Discrete Algorithms, SODA 2006, 117-123, 2006.
17. M. Kacprzak, A. Lomuscio and W. Penczek. Verification of multiagent systems via unbounded model checking. In AAMAS-04, 638-645, 2004.
18. B. P. Kooi. Probabilistic Dynamic Epistemic Logic. *Journal of Logic, Language and Information* 2003, 12: 381-408.
19. K. L. McMillan. Symbolic model checking: An Approach to the State Explosion Problem. Kluwer Academic, 1993.
20. B. Milch and D. Koller. Probabilistic Models for Agent's Beliefs and Decisions. Proc. 16th Conference on Uncertainty in Artificial Intelligence 2000: 389-396.
21. I. Walukiewicz. Completeness of Kozen's axiomatisation of the propositional  $\mu$ -calculus. *Information and Computation* 157: 142-182, 2000.
22. M. Wooldridge, M. Fisher, M. Huget, and S. Parsons. Model checking multiagent systems with mable. In AAMAS-02, 952-959, 2002.

# A Task Management Architecture for Control of Intelligent Robots

Jaeho Lee and Byulsaim Kwak

Dept. of Electrical and Computer Engineering, The University of Seoul  
90 Cheonnong-dong, Tongdaemun-gu, Seoul 130-743, Korea  
jaeho@uos.ac.kr, semix2@naver.com

**Abstract.** Designing and building Intelligent robots involves integration of various functionalities such as manipulation, navigation, various recognitions, speech understanding and expression, reasoning, planning, and so on. Furthermore, such functional components are often distributed over several processors even in a single robotic system. Manifold functionalities and inherent complexity of robotic systems require a well-organized uniform control of the functional components so that the formidable integration of the functionalities becomes manageable. In this paper, we present an agent-based task management architecture for control of intelligent robots to simplify the integration task. The task manager works as an integration middleware and provides a consistent and unified control view for the functional components which may be distributed over a network.

**Keywords:** Intelligent Robot, System Integration, Middleware, Agent Architecture.

## 1 Introduction

The intelligent robot refers to a mobile robot that can voluntarily manipulate things with its own perception and cognition of external circumstance. Intelligent service robots should be able to use their computational, sensory, and physical resources to achieve goals and respond to changes in the environment [1]. As an intelligent system, intelligent robots are also characterized by their performance over time, and their ability to evolve and learn. While conventional industrial robots typically perform pre-programmed and routine manufacturing jobs for humans. Service robots, on the other hand, serve and interact with human in the daily lives by providing services relevant to the situation.

Designing and building Intelligent robots thus involves integration of various functionalities such as manipulation, navigation, various recognitions, speech understanding and expression, reasoning, planning, and so on. Furthermore, such functional components are often distributed over several processors even in a single robotic system. Manifold functionalities and inherent complexity of robotic systems require a well-organized uniform control of the functional components so that the formidable integration of the functionalities becomes manageable.

In this paper, we present an agent-based task management architecture for control of intelligent service robots to simplify the integration task in our agent-based framework for integrated knowledge-based reasoning and task management for intelligent service robots in dynamic environments. The TASK MANAGER (Section 2) works as an integration middleware and provides a consistent and unified control view for the functional components which may be distributed over a network. Basically, the framework is designed following the agent model. An agent is a software system that is situated in an environment and that continuously *perceives, reasons, and acts* [2].

## 1.1 Motivation

The concept of an agent as an autonomous system, capable of interacting with other agents in order to satisfy its design objective has led to the growth of interest within software designers as a new software engineering paradigm [3]. Agent-based systems are increasingly being applied in a wide range of areas including business process modeling, telecommunications, computer games, military simulations, and intelligent robots. The agent paradigm has an enormous influence on model building and simulation because of the natural capacity of multi-agent systems to design and simulate complex systems. Multi-Agent systems that are composed of collections of autonomous, interacting agents offers great means to model and simulate real worlds with complex, dynamic, and nondeterministic environments where agents need to function with exogenous events and uncertain effects as well as other agents. Our TASK MANAGER borrows the concept and methodology of multi-agent systems since intelligent robots as autonomous systems have the following properties:

**Heterogeneous multiple components:** Intelligent robots require integration of multiple functional components such as manipulations, navigations, recognitions, speech understanding and expression, reasoning, planning, and so on. Those components are furthermore heterogeneous in terms of implementation languages and platforms.

**Distributed components:** The platform for an intelligent robot typically has multiple Single Board Computers (SBCs) to provide enough processing power to meet the real-time requirements and to properly handle parallel and layered robot activities.

**Intelligent systems:** Intelligent robots are inherently intelligent systems. According to Erman [4] and recitation by Booch [5] in the context of software engineering, intelligent systems differ from conventional systems by a number of attributes as follows:

- *They pursue goals which vary over time.*
- *They incorporate, use, and maintain knowledge.*
- *They exploit diverse, ad hoc subsystems embodying a variety of selected methods.*
- *They interact intelligibly with users and other systems.*
- *They allocate their own resources and attention.*



**Evolving systems:** Service robots are one type of intelligent machine that is designed to interact with human users in relatively unstructured environments [1]. A robot is also commonly defined as “A reprogrammable, multi-functional manipulator designed to move material, parts, tools or specialized devices through various programmed motions for the performance of a variety of tasks [6],” emphasizing the “programmable” or “reprogrammable” aspects of robots. Especially the performance goals for an intelligent robot system are continually increasing in complexity. This suggests that the robot architecture should be designed to evolve over time and have new capabilities added to it.

Conventional control systems and architectures are no longer adequate to realize the characteristics of the intelligent systems completely, nor are they sufficient to master the complexity of such systems. Our approach is to design the system as a multi-agent systems by ascribing to selected components ‘autonomous systems’ which can act mainly on themselves without external control most of the time. We model the autonomous systems as agents since an agent is a self-contained, concurrently executing thread of control that encapsulates some state and communicates with its environment and possibly other agents via some sort of message passing [7]. Higher degree of flexibility can then be achieved based on a control system consisting of such highly independent modules composed to fit for an application. In our agent-based control system, the resources as well as the agents are designed to operate as autonomous subsystems. Resources are mostly passive or simple active units such as manipulable objects or grippers.

Control of a complex system like intelligent robots can efficiently be realized by successively delegating control competence in hierarchically structured, autonomously operative modules, each of which implements a manageable part of the whole system. TASK MANAGER, the top-most controller of the system, as shown in Figure 1 is especially built using the BDI model. BDI agents [8] are modeled in terms of beliefs, desires and intentions and believed to provide sufficient level of abstraction to overcome the complexity of intelligent robots (see Section 2.1).

## 1.2 Objectives

There is no single perfect architecture to embody intelligence for intelligent robots. Our experience suggests, however, the most important characteristic is the ability to handle the complexity of the system and exploit diverse evolving capabilities since the essence of intelligent behavior is the structured integration of various functional components in a simple and unified system. Our objectives in designing and building a control structure for intelligent robots are summarized as follows:

- *Simple and unified control for heterogeneous components:* Integration of multiple functional components that are developed independently requires the control interface between the components and the controller to be simple

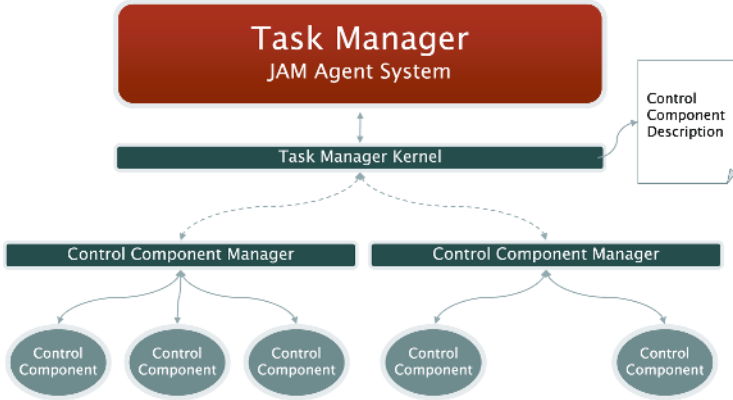
and unified. The control interface should be also independent of the language that is used to build the components and the platform where the components run. Ontological commands using command messages instead of application program interfaces (APIs) are much desired to achieve these ends because a single simple API can be used for diverse commands that are agreed upon the ontological commitment.

- *Location transparency*: A common issue of distributed software systems is the allocation of the logical software components of a system to physical nodes. In a distributed system, location transparency allows the resources to be accessed by a user anywhere on the network without concerning where the resource is located. Location transparency is the ability to map a single specification to a number of different physical environments without requiring changes to the code. Likewise, the various components located in different SBCs need to be accessed without prior knowledge of their location and to operate independently of their locations. The actual locations of components, however, need to be specified in explicit configuration of the system to support location transparency.

Location transparency eliminates location as a concern for the developer and the need to overlap of development processes among design, development and the deployment phase. This is likely to produce benefit in the development of intelligent robots because various components are independently developed on their own platform to be integrated later in the development phase.

- *High-level control*: As discussed in Section 1.1, intelligent robots as intelligent systems are complex systems that should be able to use their computational, sensory, and physical resources to achieve goals and respond to changes in the environment. Such complex systems need higher level of control than the level for the conventional systems.

The *perceive-reason-act* cycle of agent-based approaches combined with the agent-based communication using ACL messages seem to provide an appropriate level of abstraction for high-level control of intelligent robots. An agent model provides a conceptual *boundary* between the interior and the exterior. The *interior* of the boundary gives freedom of being independent from other agents or being autonomous. The *exterior* of the boundary provides a sufficient level of abstraction to other agents so that the complexity of interactions between objects can be reduced by means of the agreement or protocol of interaction between agents. This modularity provided by the notion of agent allows us to develop components independently and easily integrate into the intelligent system. The integration does not require any modification of the system since it uses predefined *goal-based interaction* instead of interdependent function calls or direct method invocation as in the object-oriented methodology. The goal-based interaction also provides additional benefit for the intelligent systems. Function calls or method invocations in the object-oriented programming tend to generate deterministic outcome. On the other hand, in our agent-based approach, the desired effect is given to each agent in the form of goals and thus agents are free to



**Fig. 1.** TASK MANAGER Architecture

choose whatever suitable course of actions to achieve the goal adapting to the dynamically changing situation.

## 2 The Architecture

The overall robot control architecture of our intelligent robot adopts the traditional three-layer architecture [9] that consists of three components: a reactive feedback control mechanism (reactive layer), a mechanism for performing time-consuming deliberative computations (deliberative layer), and a reactive plan execution mechanism (sequencing layer) that connected the first two layers. These components run as separate computational processes as this is essential for three-layer architectures.

The most important component concerning the structure of our architecture is the TASK MANAGER(TM) which resides in the deliberative layer of the underlying three-layer robot architecture. The TM provides interface to the CONTROL COMPONENTS(CC)s in the sequencing layer through TASK MANAGER KERNEL(TMK). The TM also controls the operations of the modules responsible for deliberative computations such as time-consuming planning and high level reasoning. An overview of our TM architecture is shown in Figure 1. CONTROL COMPONENTS(CC)s are components which are designed to interact with the TM. The CONTROL COMPONENT MANAGER(CCM) is a daemon process on each machine to manage CCs to be deployed on that machine. It uses agent communication language (ACL) to communicate with TM via TMK. It dynamically loads or unloads CCs on request of the TM and delivers TM's commands to CC and CC's notification of events or queries to the TM. The CCM combined with TMK works as a kind of agent middleware for the CC agents and the TM agent. TM uses the explicit XML specification of the CCM and CC as shown in Figure 2 to access and control the CCs. In the following sections, we discuss the details of the architectural building blocks.

## 2.1 Task Manager(TM)

TASK MANAGER(TM) is an agent with goals, plans, and world model that commands and controls the CONTROL COMPONENTS(CCs) based on the contexts and deliberative planning and reasoning. TM is currently implemented as a JAM [10] BDI agent. BDI agents [8] are modeled in terms of beliefs, desires and intentions. The BDI architecture underlying the implemented systems such as UM-PRS [11], JAM, and dMARS [12] is quite general. It comprises a belief base, a set of current desires or goals, a plan library and an intention structure. The plan library provides plans describing how certain sequences of actions and tests may be performed to achieve given goals or to react to particular situations. The intention structure contains those plans that have been chosen for execution. An interpreter manipulates these components, selecting appropriate plans based on the system's beliefs and goals, placing those selected on the intention structure and executing them [13]. We believe the flexibility and efficiency of the BDI agent model is appropriate for the TM of intelligent service robots and our experience with the real service robots suggests the effectiveness of our approach for TM(see Section 3).

## 2.2 Task Manager Kernel(TMK) and Control Component Manager(CCM)

As a part of the agent middleware between CCs and the TM, TMK in the deliberative layer delivers commands from the TM to a CC via the CCM. It is also a channel for notification of events or query from CC to TM. TMK maintains CC description to assist the CCM to load CCs dynamically into memory as requested by the TM. The CC description contains information written in XML on CCM and CC. The information is maintained by TMK and delivered to the CCM as needed. Figure 2 shows an exemplary CC description. The CCM in the sequencing layer is a counterpart of the TMK to provide an interaction channel between TM and CC. It resides in a machine as a daemon process and manages the life-cycle of CCs to be deployed on that machine. It performs the LOAD command and delivers other TM commands such as START, STOP, SUSPEND, RESUME, and UNLOAD to CC. It also delivers CC's notification of events or queries from CCs to the TM. Other details of the TM commands are discussed in Section 2.4.

## 2.3 Control Component(CC)

Robot software components implement the functionalities required for the intelligent robot to perceive, reason, and acts. These components may be assembled as needed to build a CONTROL COMPONENT(CC) which has an interface to interact with the TM. Several CCs can run at the same time. TM sends commands to CCs to LOAD, START, SUSPEND, RESUME, STOP, UNLOAD, and INFORM and CC then acts accordingly as discussed in Section 2.4. CC, on the other hand, can send information on local events anytime asynchronously to the TM as an

```

<Description>
  <ControlComponentManagerList>
    <ControlComponentManager>
      <Name>TestCCManager</Name>
      <IPAddress>localhost</IPAddress>
      <Port>5001</Port>
    </ControlComponentManager>
  </ControlComponentManagerList>

  <ControlComponentList>
    <ControlComponent>
      <Name>TestControlComponent1</Name>
      <Classpath>tester.TestControlComponent1</Classpath>
      <Location>TestCCManager</Location>
    </ControlComponent>
    <ControlComponent>
      <Name>TestControlComponent2</Name>
      <Classpath>tester.TestControlComponent2</Classpath>
      <Location>TestCCManager</Location>
    </ControlComponent>
  </ControlComponentList>
</Description>

```

**Fig. 2.** XML Specification of CONTROL COMPONENT MANAGER and CONTROL COMPONENT

agent does. The developer of the CCs only needs to implement the following abstract interface to get such interaction enabled. The `String` argument, a message to the agent, is used to specify what to execute. Our approach is to use such a message to specify the commands uniformly instead of an API for each command.

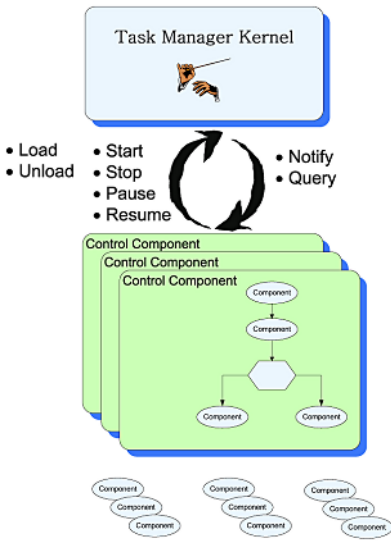
```

void execute(String XMLParameter);
CC runs this method when TM commands to START.

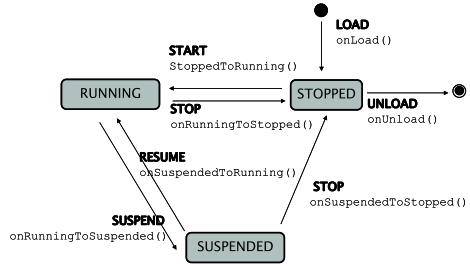
```

## 2.4 Task Manager Commands

CC acts according to the commands that are received from the TM. CC can *notify* TM of new events asynchronously, that is, inform TM of any information anytime asynchronously. CC also can *query* the status of the facts maintained in the world model of the TM. This relationship is depicted in Figure 3. The command from TM causes a state transition of CC. CC will be one of the states among `Stopped`, `Running`, or `Suspended` as shown in Figure 4. The following is the list of commands that TM can send to CCs (or CCM in case of the `LOAD` command).



**Fig. 3.** Interaction of TASK MANAGER and CONTROL COMPONENTS



**Fig. 4.** States and State Transitions of a CONTROL COMPONENT

```

void onStoppedToRunning();
void onRunningToStopped();
void onRunningToSuspended();
void onSuspendedToRunning();
void onSuspendedToStopped();
    
```

**Fig. 5.** States Transition Functions of a CONTROL COMPONENT

- **LOAD:** CCM loads the specified CC into memory. The CC is then initialized and the state of the CC is set to **Stopped**.
- **UNLOAD:** The specified CC stops and unloads itself from the memory. No state of the CC needs to be maintained.
- **START:** The specified CC runs the void execute(XML parameter) method. The CC changes state to **Running**.
- **SUSPEND:** The specified CC pauses until the RESUME command is received. The CC changes state to **Suspended**.
- **RESUME:** The specified CC resumes its operation from the suspended state. The CC changes state to **Running**.
- **STOP:** The specified CC stops its operation. The CC changes state to **Stopped**.

Just before the transition of states from **XXX** to **YYY**, the CC automatically calls the method `OnXXXTToYYY()`. The possible state transition functions are listed in Figure 5. We found the state transition functions are highly handy for the developer of CC to specify sophisticated behaviors based on state transitions. Voluntary release of shared resources as the CC changes states to **Suspended** or regaining of them as it resumes can be easily specified in `onRunningToSuspended()` and `onSuspendedToRunning()` respectively. Similarly, the action to be performed as the CC is loaded or unloaded can be specified in the void `onLoad()` and void `onUnload()` respectively as follows:

```
void onLoad();
```

This method is invoked automatically right after the CC is loaded into memory. The state of CC is set to **Stopped**.

---

```
void onUnload();
```

This method is invoked automatically right before the CC is unloaded from memory.

The actual change of states of CC or fulfillment of the commands from the TM happens only when the CC calls the `tmCheckPoint()` method. In other words, CC needs to call this method from time to time to ensure the commands from the TM are checked and honored properly by the system. This is a way of specifying an atomic behavior in our TM architecture. The code in between two `tmCheckPoint()` invocations is atomic in that it runs to the end without being interrupted.

```
void tmCheckPoint();
```

The CC calls periodically to check and execute the commands from the TM. The CC state is changed when this method is invoked.

The CC can use the `tmNotify(String XMLMessage)` method to notify TM of new events, or to inform TM of any information anytime asynchronously.

```
void tmNotify(String XMLMessage);
```

CC send to TM information that is specified as an XML message. The content of the message is ontologically defined beforehand. This method can be invoked anywhere anytime in the CC.

### 3 Experiments

Our TM was successfully applied to “T-Rot,” a bartender robot (Figure 6) at the seven-day IT exhibition at the BEXCO convention center on the sidelines of the Asia-Pacific Economic Corporation (APEC) summit in Pusan, South Korea, November 2005. T-Rot has been developed as a service robot for the elderly and disabled and has the ability to hold conversations with people and fetch items that they want. The T-Rot is equipped with two cameras that recognize people it lives with and objects such as refrigerators, glasses and other items in the living room. It also recognizes its own location, understands human speech and carries on appropriate conversations in predefined domain.

The T-Rot has about ten CCs such as navigation, arm manipulation, gripper control, face recognition, speech recognition, gesture recognition, audio expression, speech expression, world modeling, multi-modal interface, and so on. Each



Fig. 6. Robot Café



Fig. 7. T-Rot tending bar at Robot Café <sup>2</sup>

©2003 DIGITAL CHOSUN. <http://english.chosun.com/w21data/html/news/200511/200511150007.html>

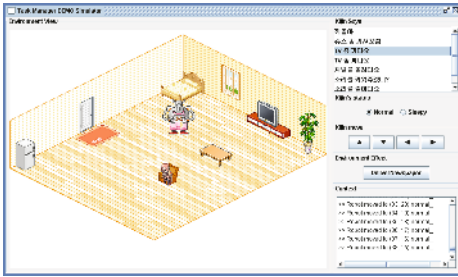


Fig. 8. TASK MANAGER Simulator

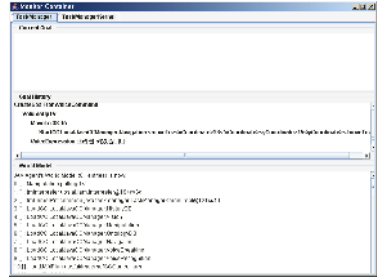


Fig. 9. TASK MANAGER Monitor

CC is again composed of multiple software components. In this demonstration, integration in a short time, say, a month, of such many components that had been developed independently was the crucial factor of success. The simplified and unified agent-based integration framework that is provided by the TM architecture enabled successful demonstration in time.

We also developed a simulator (Figure 8) and a monitor (Figure 9) for the TM to assist both the developers of CCs to test their CCs independently of other's CCs and the system integration team to integrate partly available CCs. In summary, we evaluate our achievement as we established the followings:

- Complete vertical system integration for three-layer robot architecture over reactive, sequencing, and deliberative layers.
- Clean and compact interfaces for system integration with minimal requirements for CC implementations and fine controls based on states and state transitions.
- Agent-based two-way interactions between TM and CCs with both asynchronous commands from the TM to CCs and asynchronous notification to the TM from CCs.



- Flexible ontological data interface using ACL messages and ontological definitions of commands and information.
- Reactive and proactive task management based on both reactive and proactive TM agent architecture.

## 4 Summary and Future Work

In this paper, we presented a novel way of integrating software components for intelligent robots using the abstract agent model of interactions. Through real-world applications that have been successfully demonstrated, we were able to analyze the strengths, weaknesses, opportunities, and threats (SWOT) as shown in Figure 10.

<b>Strengths</b>	<b>Weaknesses</b>
<ul style="list-style-type: none"> <li>- Simple and clean Interface.</li> <li>- Flexible open architecture.</li> <li>- Intelligent task management based on BDI agent.</li> <li>- Asynchronous command and notification mechanism.</li> <li>- Platform and language independence.</li> </ul>	<ul style="list-style-type: none"> <li>- Serious stress test yet required.</li> <li>- Communication overhead caused by the CCM and the TMK.</li> <li>- Horizontal integration among CCs in a layer as well as the vertical integration over layers.</li> </ul>
<b>Opportunities</b>	<b>Threats</b>
<ul style="list-style-type: none"> <li>- Multiple Single Board Computers (SBCs) in a robot demanding interaction middleware.</li> <li>- Component-based developments of robot software to compose CCs.</li> <li>- Standards for XML-based data interface and ontological specification.</li> <li>- Service Oriented Architecture with supporting tools and standards for interoperability.</li> </ul>	<ul style="list-style-type: none"> <li>- Requirement for rigorous analysis to prevent or resolve undesired states such as deadlocks, livelocks, and starvation.</li> <li>- Lack of ontologies for data integration and the cost of ontology engineering.</li> <li>- Supports for the non-compliant legacy components.</li> </ul>

**Fig. 10.** SWOT Analysis

Our future work will focus on exploiting the opportunities while reinforcing areas of the identified weakness, and overcoming the threats. Especially, we believe that our TM architecture is general and flexible enough to be applied in the area of ubiquitous computing and general distributed computing applications.

**Acknowledgments.** This work was supported by the University of Seoul in 2005.

## References

1. Pack, R.T.: IMA: The Intelligent Machine Architecture. PhD thesis, Vanderbilt University, Nashville, Tennessee (2003)
2. Russell, S.J., Norvig, P.: *Artificial Intelligence : A Modern Approach*. Prentice Hall (1995)
3. Jennings, N.R., Sycara, K., Wooldridge, M.: A roadmap of agent research and development. *Autonomous Agents and Multi-Agent Systems* **1** (1998) 7–38
4. Erman, L.D., Lark, J.S., Hayes-Roth, F.: ABE: An environment for engineering intelligent systems. *IEEE Transactions on Software Engineering* **14**(12) (1988) 1758–1770
5. Booch, G.: *Object-Oriented Analysis and Design*. Addison-Wesley (1994)
6. Dowling, K.: Robotics: comp.robotics frequently asked questions. <http://www.faqs.org/faqs/robotics-faq/> (1995)
7. Wooldridge, M., Jennings, N.R.: Intelligent agents: Theory and practice. *The Knowledge Engineering Review* **10**(2) (1995) 115–152
8. Rao, A.S., Georgeff, M.P.: BDI agents: From theory to practice. In: *Proceedings of the First International Conference on Multiagent Systems*, San Francisco, California (1995) 312–319
9. Gat, E.: On three-layer architectures. In Kortenkamp, D., Bonnasso, R.P., Murphy, R., eds.: *Artificial Intelligence and Mobile Robots: Case Studies of Successful Robot Systems*. MIT/AAAI Press, Cambridge (1997) 195–210
10. Huber, M.: JAM: A BDI-theoretic mobile agent. In: *Proceedings of the Third International Conference on Autonomous Agents (Agents '99)*, Seattle, Washington (1999)
11. Lee, J., Huber, M.J., Durfee, E.H., Kenny, P.G.: UM-PRS: an implementation of the procedural reasoning system for multirobot applications. In: *Conference on Intelligent Robotics in Field, Factory, Service, and Space (CIRFFSS '94)*, Houston, Texas (1994) 842–849
12. d’Inverno, M., Kinny, D., Luck, M., Wooldridge, M.: A formal specification of dMARS. In: *Agent Theories, Architectures, and Languages*. (1997) 155–176
13. Schroeder, M., Wagner, G.: Vivid agents: Theory, architecture, and applications. *Applied Artificial Intelligence* **14**(7) (2000) 645–675

# Multi-agent Based Selfish Routing for Multi-channel Wireless Mesh Networks

Yanxiang He and Jun Xiao

College of Computer, Wuhan University, Wuhan, P.R. China  
yxhe@whu.edu.cn, hugesoftcn@yahoo.com.cn

**Abstract.** Agent based systems are being used to solve problems ranging from Web search strategies to autonomous robots. In this paper we discuss a multi-agent based selfish routing problem in multi-channel wireless mesh networks. In this selfish routing,  $k$  pairs of source-sink (destination) pairs are managed by  $k$  different intelligent agents. All the agents are selfish; each agent's goal is to minimize its own cost without considering the overall performance of the network. Meanwhile, the agents are rational, they always meet the embarrassment that whether they face the probability of causing interference for routing flow by a short cut or they route flow by a longer way steadily. We introduce the game theoretic tools to direct the agents to make feasible choices. We yield a sufficient condition of the existence of pure strategy Nash equilibria and proof it in this paper (we call this the *Strong Transmission Game*). Some simulations reveal the feasibility of our proposition.

**Keywords:** Multi-Agent, Wireless Mesh Networks, Game Theory, Nash Equilibrium, Selfish Routing.

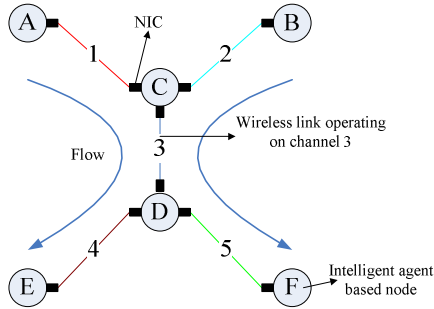
## 1 Introduction

Generally speaking, if agents are supposed to be adaptive ones, they should cast their action spaces in a form that is as amenable as possible to machine learning techniques, which is complex to be put in practice. But if the agents are supposed to be adaptive to selfish routing, the situation is much different because the agents can deal with this problem by game theoretic tools, which is exactly what we are going to introduce in this paper.

Unlike traditional wireless networks, multi-radio and multi-channel are helpful to alleviate interference among multiple simultaneous transmissions in wireless mesh network [4, 5]. It makes the transmission in Figure 1 can work simultaneously if there are more than 5 distinct channels and more than 3 radios on node  $C$  and node  $D$  respectively in this multi-agent based wireless network.

Here, we study selfish routing problem in multi-radio multi-channel wireless mesh networks, where the intelligent agents get to choose their paths and make channel assignments for their transmissions. Unlike most other work on this topic, we assume that the source nodes are owned by different entities and controlled by different agents, whose goals are to minimize their own costs of routing without considering the overall cost of the network. We emphasize the collision here is about the

probability of the interference that is generated by intelligent agents for their irrational selfish choices. Game theory is an appropriate tool to study such a routing. If all the agents can analyze the other's choices and converge to a pure strategy Nash equilibrium, it will be a feasible ending for all of them [11].



**Fig. 1.** A wireless Mesh network with 6 nodes

Papadimitriou pointed out *If the Internet is the next great subject for Theoretical Computer Science to model and illuminate mathematically, then Game Theory, and Mathematical Economics more generally, are likely to prove useful tools* [2]. In fact the game theory is able to implement the allocation scheme in a distributed manner with minimal communication overheads [3]. We believe that the interactions of the network nodes, especially on the data link layer and the network layer, should be studied as a game. On the data link layer, selfish agents would like to abuse the protocol for fixing transmission radius and the protocol of assigning channel resources; on the network layer, the emphasis is about saving the energy for the selfish agents, which is not an important point for wireless mesh networks [1]. In this paper, we will focus on the data link layer: we tried to solve the multi-agent based multi-channel selfish routing problem in a distributed manner using game theoretic tools on the data link layer. Throughout this paper, we will use sinks and destinations interchangeably, as well as routes and paths, edges and links. Our contributions are as follows.

- ◆ We present an appropriate definition of channel interference which can help these intelligent agents to avoid interference precisely.
- ◆ We formulate the multi-agent based selfish routing problem as a *Strong Transmission Game*; this game is a special case, where each agent can transmit flow to corresponding sink node simultaneously without interference among them.
- ◆ We introduce the theorem about the existence of pure strategy Nash equilibria in the *Strong Transmission Game* and bring forward our heuristic scheme.
- ◆ We evaluate the outcome of our scheme in terms of its price of anarchy [7], the society cost and the system throughput, etc.

The rest of the paper is structured as follow. The related works are presented in Section 2. We define this problem and discuss the existence of pure strategy equilibria in Section 3. We give our heuristic scheme in Section 4. We evaluate our scheme through simulations in Section 5 and draw a conclusion in Section 6.

## 2 Related Works

Some researches focused on the routing problem based on agent architecture. Sam R. Thangiah et al. introduced an agent architecture for solving vehicle routing problems by a distributed search scheme [23]. David H. Wolpert et al. took both machine learning techniques and Stackelberg games theory into account to solve the routing problem for data networks [24]. Kwindla Hultman Kramer et al. brought the software agents techniques to wireless networks for dynamic routing [27].

On the other hand, Tim Roughgarden and Éva Tardos have widely studied the loss of network performance for the lack of regulation [8]. They found the tight upper bound of linear latency system and the bound is confessed to be  $4/3$ . T. Boulogne and E. Altman recently extended the competitive routing into multicast communications and some different criteria were applied for this type of game [14]. Adrian Vetta considered the outcome to a submodular system and analyzed the performance of approximation algorithms [6]. The above researches lead to the exploration in wireless networks. Michel X. Goemans et al. researched the market sharing games in ad-hoc networks [15]. Weizhao Wang, XiangYang Li and Yu Wang researched the truthful multicast routing in selfish wireless networks [20], Luzi Anderegg and Stephan Eidenbenz introduced a game-theoretic setting for routing in a mobile ad hoc network that consists of greedy, selfish agents [21]. Recently, Sheng Zhong, Li (Erran) Li, Yanbin Grace Liu and Yang Richard Yang studied ad-hoc games using game theoretical, cryptographic and incentive-compatible techniques [22]. All these researches induce us to consider the feasibility of using game theory to solve the multi-agent based selfish routing problem in wireless mesh networks.

## 3 Multi-agent Based Selfish Routing Problem

In this section, we will first describe our system model and notations. Then, we will formally define the *Strong Transmission Game* and prove the existence of pure strategy Nash equilibria.

We use an undirected graph  $G(V,E)$  to model the wireless mesh network where  $V$  is the set of  $n$  wireless stationary nodes and  $E$  is the set of  $m$  edges. There are a transmission radius  $r > 0$  and an interference radius  $R = q \times r$  ( $q \geq 1$ ) associated with every node determined by the transmission power. We let  $d(u,v)$  to represent the Euclidean distance between  $u$  and  $v$ . Then there is an undirected edge  $(u,v) \in E$  connecting node  $u$  and node  $v$  if  $d(u,v) \leq r$ . The edge  $(u,v)$  in  $G$  corresponds to a wireless link between node  $u$  and node  $v$  in the wireless network. Moreover, if there is interference between  $(u,v)$  and  $(u',v')$ , it is a necessary condition:

$$d(u,u') < R \vee d(u,v') < R \vee d(v,u') < R \vee d(v,v') < R$$

Transmission may collide in two ways in wireless networks: *primary* and *secondary interference* [25]. *Primary interference* occurs when a node has to transmit and receive simultaneously or transmit/receive more than one packet on the same radio at the same time. *Secondary interference* occurs when a receiver node is just within the range of more than one sender node on the same channel.

Half-duplex operation is enforced for each radio to prevent *primary interference*. That means one radio can only transmit or receive one packet at one time. *Secondary interference* is shown as Figure 2 where the two transmissions work simultaneously on the same channel at the same time within the range of interference radius  $R$ .

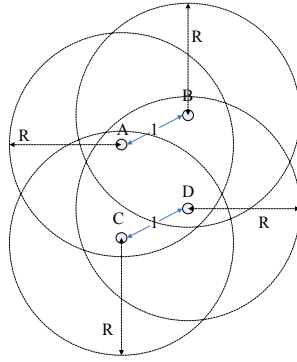


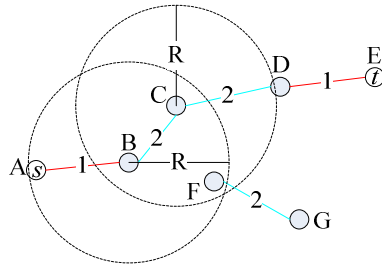
Fig. 2. An interference with 4 nodes

Here we describe the following multi-agent based selfish routing problem in wireless mesh network. There are  $k$  source-sink node pairs  $\{s_1, t_1\}, \dots, \{s_k, t_k\}$  in  $G(V, E)$ , naturally, we denote the intelligent agents set  $D = \{1, 2, \dots, k\}$  according to  $\{s_1, s_2, \dots, s_k\}$  and denote the set of  $s_i-t_i$  paths as  $P_i$  correspondingly. We let  $A_i = P_i \cup \{\emptyset\} = \{\emptyset, p_i^1, p_i^2, \dots, p_i^n\}$  to be the set of all action available to agent  $i$ , that means the agent  $i$  is in charge of choosing a path  $p_i^j$  that is from  $s_i$  to  $t_i$  with feasible channel assignment, and define  $P = \prod P_i$ . A routing flow is a function  $f : P \rightarrow \mathfrak{R}^+$ ; for a fixed flow according to agent  $i$  we denote  $f_i$  is a flow from  $s_i$  to  $t_i$ , and  $f = \bigcup_i f_i$ . The number of hops with respect to an agent  $i$  is  $h_i(f)$ . We represent the set of orthogonal channels by  $OC = \{1, 2, \dots, K\}$ ,  $|OC| = K$ . The maximum number of radios that per node can use is denoted as  $Q$  and the reality number of radios that per node use is a function  $\gamma : V \rightarrow N$ . A function  $\zeta : V \rightarrow N$  represents the number of channels that are used or interfered by a node.

*Strong Transmission Game* is a game where each agent can transmit simultaneously through a distinct path with feasible channels assignment without interference among them. It is a hard problem about channel assignment to satisfy the transmission condition to *Strong Transmission Game*. In fact [26] has proved the complexity of a channel assignment problem is NP-hard. So we will build our scheme heuristically.

In Figure 3, the edge  $BC$  interferes with  $CD$  and  $FG$ . A data package will last for three time slots to flow from node  $B$  to  $D$  (TDMA mode). In other words, the throughput is reduced to one third to the interference free one. The cost is trebled on the contrary. We use  $IE(e)$  to denote the number of links that interfered with link  $e$  on

an arbitrary channel and  $IE(e,i)$  to denote the number of links that interfered with link  $e$  on channel  $i$ . Taking the consideration in Figure 3 into account, we define the private cost function to agent  $i$  as  $c_i(f) = h_i(f) \times \max_{e \in p_i} IE(e,i)$  and the social cost function  $\kappa(f) = \sum c_i(f)$ . As the intelligent agents are selfish and rational, they will minimize their private costs and take the interference into account. That means an agent will minimize its  $c_i(f)$  subject to  $\gamma(v) \leq Q \quad \forall v \in p_i^k$  and  $\zeta(v) \leq K \quad \forall v \in p_i^k$ . What we want here is to know the deteriorative degree that is caused by the agents' selfish.



**Fig. 3.** A flow demonstration

We say that action profile  $P \in P$  is a pure strategy Nash equilibrium if no agent has an incentive to change its action. That is, for any agent  $i$ ,

$$c_i(f(P)) \leq c_i(f(P'_i))$$

Here  $P'_i$  represents the agent  $i$  change its action, which is different from the one in  $P$ , to any other one in  $A_i$ .

The pure strategy Nash equilibria are important because the agents are unlikely to choose one action amongst many on the basis of probability distribution which is called as a mixed strategy Nash equilibrium. What's more, the strategy space of pure strategy ones is much smaller than the mixed strategy ones. Thus it is important to converge to pure strategy Nash equilibria. Here we will discuss what is sufficient condition for the existence of pure strategy Nash equilibria and how to converge to pure strategy Nash equilibria. By abusing the notation a little bit without confusion, we also use  $P \in P$  to denote its corresponding flow  $f$ .

**Theorem 1.** Take a *Strong Transmission Game* system  $(\kappa, c_i)$ , there is a pure strategy Nash equilibrium.

*Proof.* To prove this Theorem is just to find a function that can reach a pure strategy Nash equilibrium. Consider a directed graph  $D$ , whose node corresponds to one of the possible pure strategy profile which is feasible path and corresponding channel assignment set here. There is an arc from node  $P_0 = \{p_1, \dots, p_i, \dots, p_k\}$  ( $p_i$  represents the path and the channel assignment according to agent  $i$ ) to node  $P_1 = \{p_1, \dots, p'_i, \dots, p_k\}$  if  $c_i(P_1) < c_i(P_0)$  for some agent  $i$ . It follows that a node  $P_N$  in  $D$  corresponds to a pure

strategy Nash equilibrium if and only if the node has out-degree zero. So if  $D$  is acyclic and there is a node that can satisfy the flow simultaneously, the system has a pure strategy Nash equilibrium. The existing of the flow is satisfied from the *Strong Transmission Game* proposition; we will prove that  $D$  is acyclic below.

Suppose  $D$  is not acyclic. Then take a directed cycle  $C$  in  $D$ . Suppose the cycle contains nodes corresponding to the path sets:  $P_0 = \{p_1^0, p_2^0, \dots, p_k^0\}$ ,  $P_1 = \{p_1^1, p_2^1, \dots, p_k^1\}, \dots, P_i = \{p_1^i, p_2^i, \dots, p_k^i\}$ , Here  $P_0 = P_i$ . It follows that the action profile  $P_r$  and  $P_{r+1}$  differ in only the path of one agent, say agent  $i_r$ . So  $p_i^r = p_i^{r+1}$  if  $i \neq i_r$  and  $c_{i_r}(P_{r+1}) < c_{i_r}(P_r)$ , it must be the case that  $\sum_{r=0}^{i-1} c_{i_r}(P_{r+1}) - c_{i_r}(P_r) < 0$ . But because in order to avoid interference, if the intelligent agent  $i_r$  changes its path or adjusts channel assignment, the change will not interfere with the other agents' choices of their paths and channel assignments. That means  $c_{i_r}(P_{r+1}) - c_{i_r}(P_r) = \kappa(P_{r+1}) - \kappa(P_r)$ . Then, since  $P_0 = P_i$ , we obtain

$$\begin{aligned} \sum_{r=0}^{i-1} c_{i_r}(P_{r+1}) - c_{i_r}(P_r) &= \sum_{r=0}^{i-1} \kappa(P_{r+1}) - \kappa(P_r) \\ &= \kappa(P_i) - \kappa(P_0) \\ &= 0 \end{aligned}$$

That is a contradiction to  $\sum_{r=0}^{i-1} c_{i_r}(P_{r+1}) - c_{i_r}(P_r) < 0$ . So  $D$  is acyclic. Then Theorem 1 follows.  $\square$

But it is a well-known open problem to compute Nash equilibria in an arbitrary multi-player game [10, 19], except some special ones [12, 13]. Then we will give our heuristic algorithm in the next section.

## 4 Routing and Channel Assignment

In this section, we will introduce a method for each agent to find a distinct route on a distinct channel firstly, which in fact can converge to a pure strategy Nash equilibrium if the channel assignment is not taken into account.

Here we assume the orthogonal channels are more than the source-sink node pairs, i.e.  $K \geq k$ . Thus the agents can begin their exploration of distinct paths on distinct channels. We assume that each node uses two different radios to receive and send data simultaneously without interference on two different channels, the throughput of the routing will be doubled to the single radio one. Here we assume each node has even radios (2, 4, 6 ...), which is a necessary condition to the high throughput transmission. On the other hand, we prefer a path with relatively low hop-count because the more hops a candidate path has, the longer delay packets will suffer, and more importantly, the higher bandwidth invalidation probability it will cause. Therefore, each agent tries to find a minimum hop-count maximum radio utility path as the selected path for itself. Here we present an algorithm to find such a path for an agent.



**Algorithm 1.** Minimum hop-count maximum radio utility path algorithm

- 
- Step\_1 Construct an auxiliary graph  $G_{s-t}(V_{s-t}, E_{s-t})$  where  $V_{s-t}$  contains wireless nodes, which have even spare radios except node  $s$  or  $t$ , and  $E_{s-t}$  contains all links incident with  $V_{s-t}$ .
- Step\_2 Apply Breadth First Search (BFS) algorithm to compute an  $s-t$  path  $p$  in  $G_{s-t}$  with minimum hop-count.
- Step\_3 Assign all the link  $e \in p$  with the same spare channel and update the number of spare radios on mediate  $s-t$  nodes by subtracting 2 from the foregoing number.
- 

The Algorithm 1 gives a function to find a feasible path for an agent; we introduce the Algorithm 2 to give a function to each agent to find a feasible path with a view to the interference among them.

**Algorithm 2.** Interference-aware Minimum hop-count maximum radio utility path algorithm

- 
- Step\_1 Initialize the agents array  $D = \{1, 2, \dots, k\}$  and paths set  $P = \phi$ .
- Step\_2 **for** each agent  $i \in D$   
     Use **Algorithm 1** on agent  $i$  to find a minimum hop-count path  $p_i$ .  
     **if** path  $p_i$  is feasible  
          $P = P \cup p_i$ .  
     **endif**  
**endfor**
- 

Now we have assigned each agent a feasible path and then we will use overall channels to improve each agent's performance next. We use notations *inter* and *intra* to represent the interference that caused by the link from other agents' transmission path and the link from itself respectively. The algorithm is below.

**Algorithm 3.** Channel assignment algorithm

- 
- Step\_1 Initialize the paths array  $P = \{p_1, p_2, \dots, p_k\}$ , channels set  $OC = \{1, 2, \dots, K\}$  and a number  $t$  to represent the current path, a number  $i$  to represent the current channel, a Boolean variable  $channel\_flag = \text{TRUE}$  to control the loop.
- Step\_2 **while**  $channel\_flag$   
      $channel\_flag = \text{FALSE}$   
     **for** each channel  $i \in OC$   
         **for** each path  $p_t \in P$   
             **repeat** to do  
                 Find the link  $e \in p_t$  that has the maximum  $IE(e)$  and satisfies  $IE_{inter}(e, i) = 0$  meanwhile  $IE_{intra}(e, i) = IE(e)$ . Assign this edge with channel  $i$ . Update the interfered link number of each affected link in  $G_{s-t}$  and  $channel\_flag = \text{TRUE}$ .  
             **until** none  $e \in p_t$  exists  
         **endfor**  
     **endfor**  
**end while**
-

We can see from these algorithms, there is always a baseline to be kept that no interference occurs between two agents' transmissions. This is necessary for the routing of selfish agents because no selfish agents would give way in non-cooperative game and the overall network would break down for the continuous interference. That is why we assume  $K > k$  before, and what's more, it is the key to the existence of pure strategy Nash equilibria. If the channel assignment here is regarded as another game based on the output of Algorithm 2, the Algorithm 3 can converge to a pure strategy Nash equilibrium for this separate game.

## 5 Evaluation

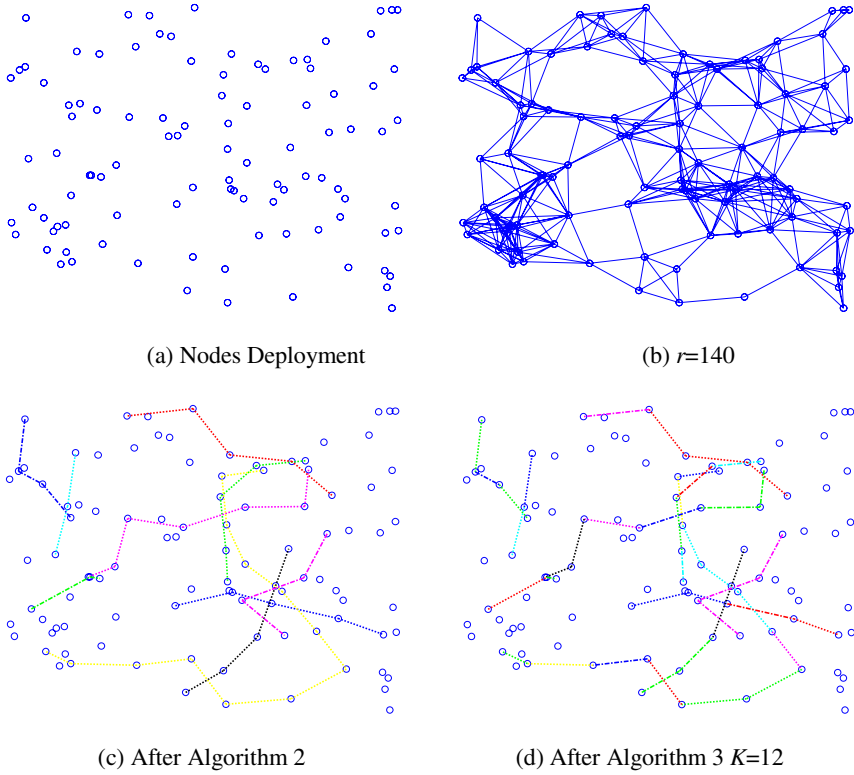
In this section we investigate the efficiency of our scheme and evaluate the affection of some factors in a simulated network scenario. We generated a wireless mesh network with 100 nodes as be shown in Figure 4-(a). These nodes are randomly placed in an  $800 \times 800$  square meter rectangular region. Each node has 2 802.11a radios whose link rate is 54 Mbps; the interference radius  $R$  is 2 times as the transmission radius  $r$ , i.e.  $R = 2r$ . We vary the transmission range and the number of  $s$ - $t$  pairs, which are generated randomly in each round, to evaluate the affection. On the other hand, the flow for each agent is set to be the same in all these simulations  $f_i = f_j, \forall f_i, f_j \in f, i \neq j$ , which will sharpen the competition among the agents. Figure 4-(b) shows the links which indicate that the distance between each two nodes incident with a link is no more than 140 meters.

From Figure 4-(c) and (d), we show an example for our algorithms where the different line colors and styles represent different channels. The  $s$ - $t$  node pairs were generated randomly to follow the given requirements. The high cost agent even can not find another feasible path to replace the current one, which proves that our scheme is feasible in this case.

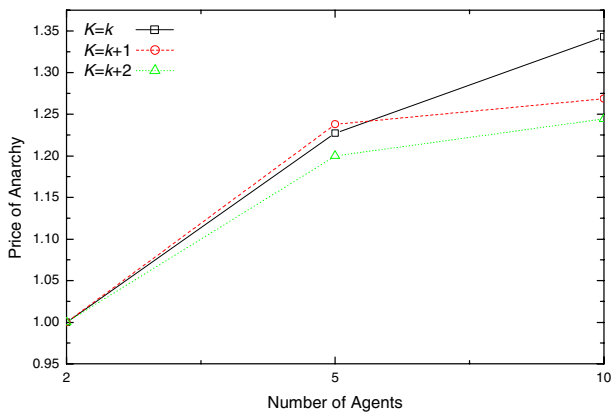
The price of anarchy is a main criterion to evaluate the efficiency of the game solution. We use a lower bound of the optimal solution instead of the optimal one to compute the price of anarchy. Here  $K$  represents the distinct channel number and  $k$  represents the  $s$ - $t$  node pairs (agents) number.

As be shown in Figure 5, the price of anarchy increases as the number of  $s$ - $t$  node pairs turns more. That shows more node pairs will sharpen the competition which induce the increase of the price of anarchy. On the other hand, as the channels turn more, the competition among these agents can be alleviated because more channels bring a lot of new mediate nodes which work on these newcomer channels. Thus the price of anarchy with more channels decreases when the agents count reaches 10.

As Figure 5 tells us the worst output of our scheme, we use Figure 6 to show some outputs that are not bad. The good outputs are the cases which exactly have optimal routing paths or optimal channel assignment; the perfect ones are just equal to the optimal output and converge to equilibria exactly. These outputs are totaled here separately and the percentage data are just in this figure. As we just use a sufficient but not necessary condition, comparing with infinite radios and channels situation, to replace the output of optimal one, so we just get the lower bounds, which induces that the exactly percentage should be bigger. Figure 5 and 6 together show that the agents will not be far from the Nash equilibrium even the optimal solution. Moreover, the perfect output percentage in Figure 6 proves the channel is a main bottleneck in the wireless mesh network.



**Fig. 4.** A network with 100 nodes and 10  $s-t$  pairs



**Fig. 5.** Price of Anarchy

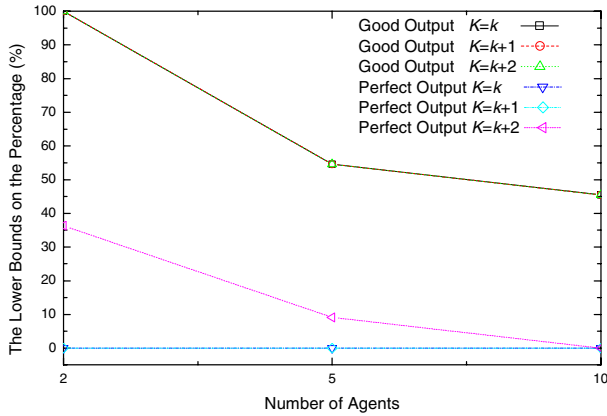


Fig. 6. The Lower Bounds on the Percentage (%)

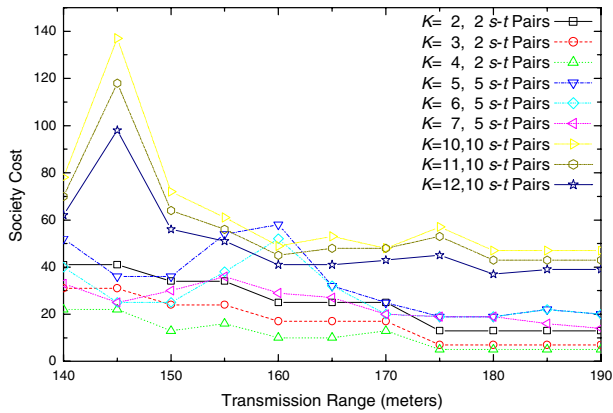


Fig. 7. Society Cost

Figure 7 reveals the relative between the transmission range and society cost with different number of orthogonal channels and agents ( $s-t$  pairs). The society cost decreases mainly as the nodes increase their transmission range. It is because that the increase of transmission power brings out some new mediate nodes for the agents which may alleviate the competition and decrease the society cost. Besides, the number of orthogonal channels is proved to be a bottleneck for routing again. When the number of orthogonal channels increases, the society cost decreases dramatically in most cases.

Figure 8 reveals the relative between the transmission range and system bandwidth with different number of orthogonal channels and  $s-t$  pairs. It is clear that as the number of orthogonal channels increases, the system bandwidth trends to near the optimal case, which proves the channel resource to be a bottleneck once and again. All these results induce that the research on the orthogonal channels would be a potential breakpoint for the multi-agent based selfish routing in wireless mesh networks.

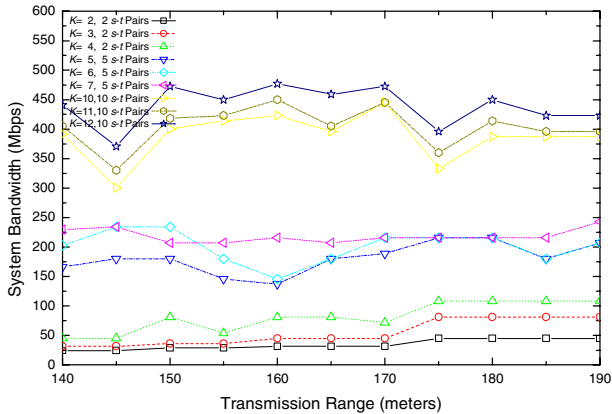


Fig. 8. System Bandwidth

## 6 Conclusion and Future Work

We introduced an attempt, which is based on multi-agent architecture, onto route flow in multi-channel wireless mesh network by using game theoretic tools in this paper; we proved the existence of pure strategy Nash equilibrium in a special game and give a heuristic scheme for agents to calculate an approximate Nash equilibrium. The evaluation shows the output of our scheme is feasible in some cases.

Recently, George Christodoulou et al. introduced the coordination mechanism design and used the price of anarchy firstly to show how much a mechanism can improve the overall system performance [9], which can greatly improve agents' reaction without much communication overhead increase. And some famous coordination mechanisms were introduced in several papers [16, 17, 18]. The theoretic conclusions prove that some perfect coordination mechanism can improve the overall system performance greatly.

An appropriate coordination mechanism is the key to improve the overall system performance. To design an appropriate coordination mechanism, which can reduce price of anarchy theoretically, is a challenge in future for us.

## References

1. IEEE 802.16-2004 Part 16: Air Interface for Fixed Broadband Wireless Access Systems, Oct. 2004.
2. C. Papadimitriou. Algorithms, games, and the internet. In *Proceedings of the 33<sup>rd</sup> Annual Symposium on the Theory of Computing*, 2001.
3. Haïkel Yaïche, Ravi R. Mazumdar and Catherine Rosenberg. A Game Theoretic Framework for Bandwidth Allocation and Pricing in Broadband Networks. *IEEE/ACM Transaction on Networking*, **8**(5), pages 667-678, 2000.
4. Mansoor Alicherry, Randeep Bhatia and Li (Erran) Li. Joint Channel Assignment and Routing for Throughput Optimization in Multi-radio Wireless Mesh Networks. In *Proceedings of ACM MOBICOM*, pages 58-72, 2005.

5. Richard Draves Jitendra Padhye Brian Zill. Routing in Multi-Radio, Multi-Hop Wireless Mesh Networks. In *Proceedings of ACM MOBICOM*, pages 114-128, 2004.
6. Adrian Vetta. Nash Equilibria in Competitive Societies, with Applications to Facility Locations, Traffic Routing and Auctions. In *Proceeding of IEEE FOCS*, 2002.
7. E. Koutsoupias and C. H. Papadimitriou. Worst-case Equilibria. In *Proceedings of the 16<sup>th</sup> Annual Symposium on Theoretical Aspects of Computer Science*, pages 404-413, 1999.
8. Tim Roughgarden and Éva Tardos. How Bad is Selfish Routing?. *Journal of the ACM*, 49(2), pages 236-259, 2002.
9. G. Christodoulou, E. Koutsoupias, and A. Nanavati. Coordination Mechanisms. In *Proceedings of the 31st International Colloquium on Automata, Languages, and Programming (ICALP)*, pages 345--357, Turku, Finland, July 2004.
10. X. Deng, C. H. Papadimitriou and S. Safra. On the Complexity of Equilibria. In *Proceedings of the 34<sup>th</sup> Annual ACM Symposium on the Theory of Computing*, pages 67-71, 2002.
11. J. F. Nash, Jr. Non-cooperative Game. *Annual of Mathematics*, 54(2): 286-295, 1951.
12. M. Kearns, M. L. Littman and S. Singh. Graphical Models for Game Theory. In *Proceedings of the Conference on Uncertainty in Artificial Intelligence*, pages 253-260, 2001.
13. J. T. Howson, Jr. Equilibria of polymatrix games. *Management Science* 18, pages 312-318, 1972.
14. T. Boulogne and E. Altman. Competitive Routing in Multicast. *Networks* 46(1), Pages 22-35, 2005.
15. Michel X. Goemans, Li (Erran) Liy, Vahab S. Mirrokni and Marina Thottan. Market Sharing Games Applied to Content Distribution in Ad-Hoc Networks. In *Proceedings of ACM MOBIHOC*, pages 55-66, 2004.
16. A. Czumaj and B. Vocking. Tight Bounds for Worst-case Equilibria. In *Proceedings of SODA*, pages 413-420, 2002.
17. M. Gairing, T. Lucking, M. Mavronicolas and B. Monien. Computing Nash Equilibria for Scheduling on Restricted Parallel Links. In *Proceedings of the 34<sup>th</sup> Annual ACM Symposium on the Theory of Computing*, pages 613-622, 2004.
18. E. Koutsoupias and C. Papadimitriou. Worst-case Equilibria. In *Proceedings of STACS*, pages 404-413, 1999.
19. Christos H. Papadimitriou. Computing Correlated Equilibria in Multi-Player Games. In *Proceedings of the 35<sup>th</sup> Annual ACM Symposium on the Theory of Computing*, pages 49-56, 2005.
20. Weizhao Wang, XiangYang Li and Yu Wang. Truthful Multicast Routing in Selfish Wireless Networks. In *Proceedings of ACM MOBICOM*, pages 245-259, 2004.
21. Luzi Anderegg and Stephan Eidenbenz. Ad hoc-VCG: A Truthful and Cost-Efficient Routing Protocol for Mobile Ad hoc Networks with Selfish Agents. In *Proceedings of ACM MOBICOM*, pages 245-259, 2003.
22. Sheng Zhong, Li (Erran) Li, Yanbin Grace Liu and Yang Richard Yang. On Designing Incentive-Compatible Routing and Forwarding Protocols in Wireless Ad-Hoc Networks-An Integrated Approach Using Game Theoretical and Cryptographic Techniques. In *Proceedings of ACM MOBICOM*, pages 117-131, 2005.
23. Sam R. Thangiah, Olena Shmygelska and William Mennell. An Agent Architecture for Vehicle Routing Problems. In *Proceedings of the 2001 ACM symposium on Applied computing*, pages 517-521, March 2001.
24. David H. Wolpert, Sergey Kirshner, Chris J. Merz and Kagan Tumer. Adaptivity in Agent-Based Routing for Data Networks. In *Proceedings of the fourth international conference on Autonomous agents*, pages 396-403.

25. S. Ramanathan and E. L. Lloyd. Scheduling Algorithms for Multihop Radio Networks. *IEEE/ACM Transactions on Networking*, vol. 1, no. 2, pages 166-177, April. 1993.
26. Ashish Raniwala, Kartik Gopalan and Tzi-cker Chiueh. Centralized Channel Assignment and Routing Algorithms for Multi-Channel Wireless Mesh Networks. *ACM Mobile Computing and Communication Review (MC2R)*, Vol. 8, no. 2, pages 50-65, 2004
27. Kwindla Hultman Kramer, Nelson Minar and Pattie Maes. Tutorial: Mobile Software Agents for Dynamic Routing. *ACM SIGMOBILE Mobile Computing and Communications Review*, Vol. 3, no 2, pages 12-16, April. 1999.

# Natural Language Communication Between Human and Artificial Agents

Christel Kemke

Department of Computer Science  
E2-412 EITC Building  
University of Manitoba  
Winnipeg, Manitoba, R3T 2N2, Canada  
ckemke@cs.umanitoba.ca

**Abstract.** The use of a complex form of language for the purpose of communication with others, to exchange ideas and thoughts, to express statements, wishes, goals, and plans, and to issue questions, commands and instructions, is one of the most important and distinguishing aspects humankind. If artificial agents want to participate in a co-operative way in human-agent interaction, they must be able - to a certain degree - to understand and interpret natural language, and translate commands and questions of a human user and map them onto a suitable set of their own primitive actions. In this paper, we outline a general framework and architecture for the development of natural language interfaces for artificial agents. We focus in this paper on task-related communication, in a scenario where the artificial agent performs actions in a co-operative work setting with human partners, who serve as instructors or supervisors. The main aim of this research is to provide a consistent and coherent formal framework for the representation of actions, which can be used for planning, reasoning, and action execution by the artificial agent, and at the same time can serve as a basis for analyzing and generating verbal expressions, i.e. commands, instructions and queries, issued by the human user. The suggested framework is derived from formal methods in knowledge representation, in particular description logics, and involves semantic and ontological descriptive elements taken from linguistics, computer science, and philosophy. Several prototypical agent systems have been developed based on this framework, including simulated agents like an interior design system and a household robot, and a speech controlled toy car as example of a physical agent.

## 1 Introduction and Overview

The core issue of this research is to provide a sound, formal framework for the representation of actions, which can be used as a basis for the development of adaptable speech and language interfaces for a variety of agent systems, including physical agents, like robots, devices like household appliances, security and other sensor-equipped systems, as well as interactive software systems, like design or construction systems.

A crucial point in the development of these systems is a representation of the domain, i.e. actions and objects, in a format, which is suitable to represent the respective contents



of natural language expressions; to provide a connection to the agent's set of actions, and to serve as a foundation for essential reasoning and inference processes.

The representational framework is based on structured concepts, which are arranged in a taxonomic hierarchy. A formal semantics defines clearly the meaning of those concept descriptions, relations between them and in particular their arrangement in the hierarchy. This representational formalism has been proposed firstly by Brachman and colleagues, exemplified in the KL-ONE knowledge representation language [5], and led in recent years to the development of Description Logics, which is a collective term for various knowledge representation languages founded on the same principle of structured concept representations in taxonomic hierarchies, with a clearly defined formal semantics [4]. Description Logic (DL) languages were conceived and used in the first instance for the representation of static concepts and their structural properties. Dynamic concepts, like actions and events, which are defined through change over time, were considered only in a few approaches, with the first work including a formal semantics of taxonomic relations and inheritance for action concepts in the mid eighties [16,17], some work addressing specific aspects of action representation in taxonomic hierarchies e.g. [3,7,27] and more recent detailed work in [8,13,14]. We adopt concepts from these works here, since it provides a suitable basis for knowledge representation, adequate for use in agent systems as well as for natural language interfaces. The linguistic representation is based heavily on case frames [10,9], since they provide a focus on the action-aspect of a natural language input, which is the crucial element for an interpretation in the context of agent systems.

In the following sections, we describe the general system architecture, the linguistic processing and the underlying ontological structure for classifying actions into a hierarchy. For further details on the formal semantics for action taxonomies, we refer to [13], and for a related semantic based classification of action concepts and discussion of the ontology to [15]; an implementation example can be found in [14].

## 2 General Architecture

The processing within the general system architecture, which connects the natural language interface to an agent system, is illustrated in Figure 1, showing the analysis and representation of a put-command in natural language as an example. This processing includes several reasoning steps, for example, the assignment of the concrete instance sofa-1 to the concept sofa, the resolution of the ambiguity of besides to left-of (as result of a clarification dialogue with the human user), and the addition and instantiation of the source-feature, in case it is needed to perform the put-action performance, to the current location of sofa-1, which can be retrieved from the knowledge base as the value of the location-feature of the specific sofa sofa-1. The result is an instantiated frame representation of the action, with references to necessary parameter-objects involved in the performance of the action. This representation can be passed over to the agent-system for execution. The agent system might still have to do further planning or reasoning processes but the action-frame is provided in such a format, that any agent system based on standard action-representations with pre-conditions and effects specified in a logic format can work with this. Other more

basic physical agents, like robots, can be connected using an additional controller or interpreter, which transforms this representation and generates actions in the specific format required by the agent or robot.

### 3 Linguistic Processing

The natural language inputs considered here, which establish the communication with agent systems, comprise typically of commands, questions, and possibly declarative statements as sentence types, as well as confirmations as part of clarification dialogues between system and user.

The natural language input is processed with a standard parsing algorithm, the Early-parser, based on a grammar adapted to these typical sentence types and structures. The structural representation of the sentence generated by the parser is then processed in a shallow syntactic-semantic analysis, which selects and determines the contents of a case-frame representation of the linguistic input, based on the sentence type and the main syntactic constituents of the sentence, like subject noun phrase, object noun phrase, prepositional phrases for locations etc. plus an indicator for the queried part of the frame in case of question-sentences. Word categories, which are stored together with the specific single words in the lexicon, are accessed and used during the parsing process.

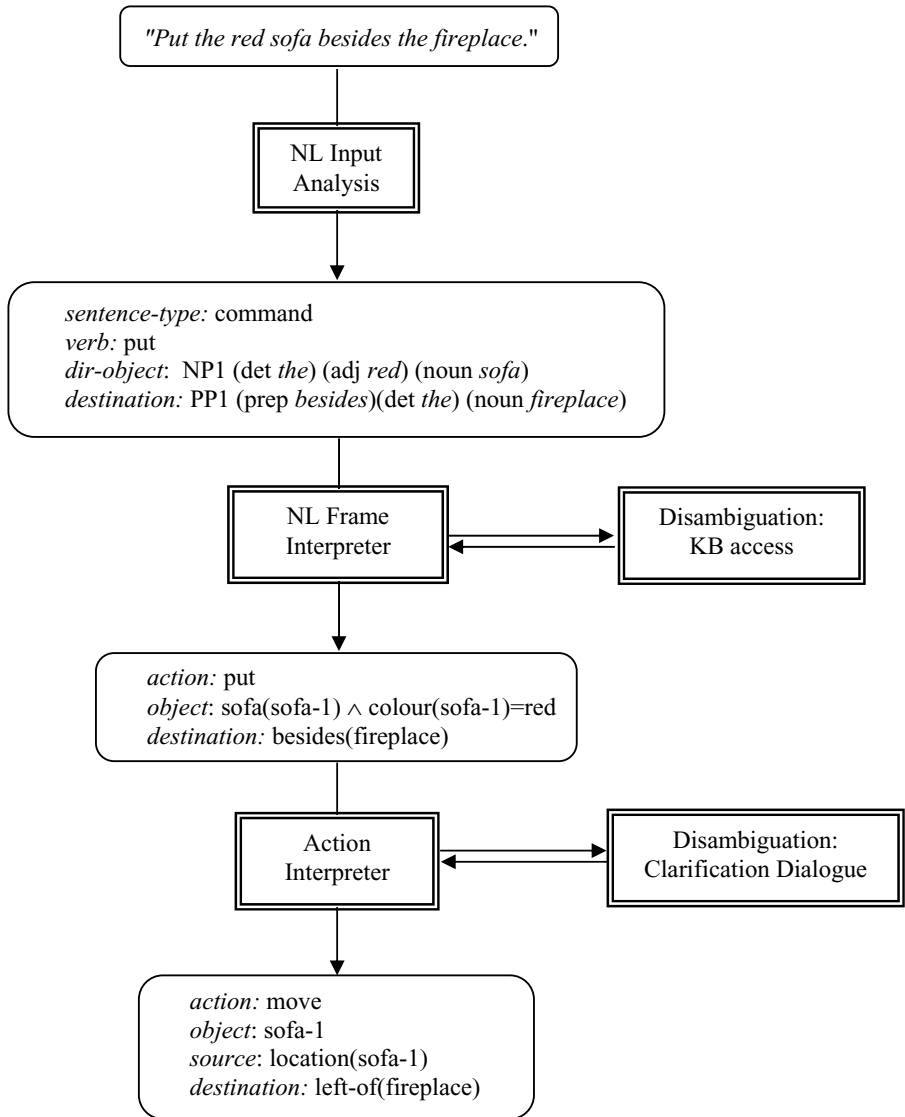
A command sentence is typically characterized through a leading verb, followed by complements to this verb, like an object noun phrase, and prepositional phrases further describing the object or action.

The lexicon provides also a connection to the knowledge base, since lexical items (words) may refer to respective concepts in the knowledge base. For example, the words "sofa" and "couch" are entries in the lexicon, which both refer to the concept *sofa* in the knowledge base.

The major part of the lexicon is constituted by verbs, nouns, attributes, modifiers, prepositions and other categories relevant to the domain of the agent system. The lexicon contains also information on necessary and optional complements to a verb, which is further taken into account in the grammar and can thus be used during parsing. For example, verbs like "put" require a direct object (what to put) as well as a destination phrase (where to put it), and verbs like "move" can have in addition a source specification (from where to move it).

Example: *"Put the red sofa besides the fireplace"*

The example above, with the simplified processing as shown in Figure 1, is taken from an interior design system we developed (see section 5.1 below). This system focuses on certain types of physical objects, like furniture, lamps, etc. and other floor plan items like fireplaces, windows, doors, and actions related to positioning and moving furniture and similar objects. Descriptions of these objects are provided through object concepts in the knowledge base, which contain also the characteristic features of such objects, which are necessary for the agent system's performance.



**Fig. 1.** Processing of a natural language command

The parser generated output comprises syntactic constituents in a still relatively "raw form", as indicated in Figure 1, but has sufficient information for the frame interpreter to provide a mapping of the linguistic representation onto a more formal, structured verb-specific action-frame representation.

This representation might still be ambiguous and allow several interpretations in the Action Interpreter, which have to be resolved. In the example in Figure 1, the red sofa has to be identified as one specific item. This can be done through accessing the

knowledge base or world model of the system. The linguistic term "besides" is also ambiguous, since it resolves to both "left-of" and "right-of". This requires a clarification dialogue with the user, where the system asks the user for the correct interpretation, or alternatively the system can pick either of the options.

## 4 Domain Representation and Ontology

The structure of the domain is internally represented as ontology, based on a knowledge representation format closely related to Description Logics [3, 8]. Classes of objects of the domain are represented in a hierarchical fashion, with general concepts, like *physical object*, as higher level nodes, and more specific concepts, like *furniture*, and even more special object classes, like *fireplace*, as lower level nodes. The hierarchical connection between those concept nodes is a pure subclass/superclass or subsumption relation, also known as "IS-A"- or "inheritance"-hierarchy. Other connections reflect relationships in the sense of roles, e.g. the *weight* of physical objects, or attributes, like their *color*.

Certain roles and attributes can be marked as fixed and not changeable, while others are modifiable and thus can form a basis and source for action specifications. For example, in an interior design system, furniture could be equipped with a non-modifiable weight-attribute but the location would be open to change, since furniture are typically being moved around, when a design process is realized in reality. Other physical objects in the domain might nevertheless still be classified as non-moveable, for example a fireplace, and their location-attribute would be qualified as persistent and not changeable. The same distinction applies to relationships between objects, where certain relations, pertaining to certain object classes can change, for example, the spatial relation *besides* can change for moveable furniture, whereas it remains persistent for non-moveable objects like windows or walls. This distinction parallels the notion of fluents in situation calculus, which are predicates whose value or extension can change over time, usually due to actions performed by an agent.

Persistent features of the domain are useful as reference points or baseline for assurances about the domain. In case of a spatially dynamic domain, for example, in which objects can be moved, fixed objects and static relations between fixed objects can always be used as reference points.

The representation of actions takes two forms of representation into account: a case frame representation, which allows an easy connection to the contents of the linguistic input, and a logic-based representation of preconditions and effects of an action, which allows a connection to planning, reasoning and action execution processes.

The format we developed for representing action concepts is based on the work described in [13-17], and integrates both aspects of actions as requested above. Actions are described in a format similar to functional specifications, with a set of parameters, which refer to object classes of the ontology, and the functionality is specified through logic formulae describing preconditions and effects, where these formulae are again based on the parameter objects, i.e. they describe changes of role or attribute values of these objects.

*put* <object> <destination>  
*object*: <moveable-object>  
*destination*: <location>

The above specifies a generic action *put* with two obvious parameters, whose values are constrained through the reference to certain object classes, i.e. *moveable-object* in the case of the object, and a *location* type in case of the destination. These object classes correspond to accordingly defined concepts in the ontology.

Furthermore, the action is specified through precondition and effect formulae:

*precond*:  $\neg\exists x. \text{ON}(x, \text{destination})$   
*effect*:  $\text{ON}(\text{object}, \text{destination})$

The occurrence of a specific *put*-action would involve the specification of the parameters, i.e. a specific object to be moved, and a specific location where to move it to, which is easily extracted from the representation of the natural language input. The reference to the object classes or concepts, which specify the possible role fillers or parameter instances, is checked when the action is instantiated.

For example, a fireplace would not be allowed as parameter to the *put*-action, since, due to the restriction that it cannot change its location-value ( $\rightarrow$  persistent *location* attribute), it is classified as *non-moveable-object*, and thus not in the class *moveable-object*. In the same way, the destination of a *put*-action has to fulfill the location criterion, thus avoiding absurd action specifications like putting a sofa on a window.

The hierarchical arrangement of objects and actions allows a structured search for proper representations and specifications of natural language descriptions and of actions of the agent system as well. Generic actions specified on a higher hierarchy level are refined on lower levels through specializing their parameters, e.g. substituting the parameter *moveable-object* of a generic *move*-action with the more specific class *furniture*, and in a further step again with the sub-concept *lamp* of *furniture*.

Thus, we derive more special action concepts through the use of more special object concepts as parameter classes. Other forms to create / find more specialized actions on lower levels in the hierarchy, include: adding parameters, for example an additional *source* for the action-concept *move*, which distinguishes it from the generic action-concepts *put* or *place*, which is qualified only through a specific destination for the object to be placed.

The use of additional parameters, like *source*, can also involve additional specifications of preconditions or effects, which is another form of specializing actions in a taxonomic hierarchy:

*move* <object> <source> <destination>  
*object*: <moveable-object>  
*destination*: <location>  
*source*: <location>  
*precond*:  $\neg\exists x. \text{ON}(x, \text{destination})$   
*effect*:  $\text{ON}(\text{object}, \text{destination}) \wedge \neg\text{ON}(\text{object}, \text{source})$

The action representations in the ontology can be described in such a way, that some level or levels of actions correspond to actions in the agent system, or can be mapped onto actions of the agent system. Since the action format defined and used in the ontological framework here includes precondition and effect representations, it allows planning and reasoning processes, and thus enables the connection to a variety of agent systems with compatible action repertoires.

## 5 Prototype Systems

Several prototype systems have been developed with different degrees of complexity regarding the natural language capabilities, and varying levels of sophistication regarding the knowledge representation and reasoning components. All these systems have in addition been equipped with a speech input and output facility, except for the toy car - which does not yet speak. Some of these systems, which have been developed mostly by students as part of course work or as extra-curricular activity, are described below.

### 5.1 Interior Design System

This system is a prototype of a combined natural language and visual scene creation system, inspired by [6]. The current implementation simulates an interior design system, which models actions and questions related to the positioning of furniture in a room, and includes the addition of furniture to the scene as well as the removal of furniture from the scene. The system works with speech input processed through the Dragon™ speech recognition system. It performs a syntactic and semantic analysis of the recognized verbal input and produces a ‘request-frame’ representing the user’s query or command. The query or command is processed through accessing a Description Logic style knowledge base.

The system is implemented in two versions, one with a knowledge base implemented in PowerLoom<sup>1</sup>, and the other one with a knowledge representation and reasoning module in CLIPS<sup>2</sup>. Both represent domain objects and actions in a conceptual framework based on inheritance hierarchies. In addition, the knowledge representation features spatial relations and respective spatial rules and constraints, like transitivity or asymmetry of spatial relations and constraints, e.g. that large objects cannot be put on top of small objects. The system also integrates commonsense rules, like a preference to specify relations in the natural language output with respect to static inventory like windows, doors or a fireplace.

### 5.2 Virtual Household Agent

One of the earliest prototype projects is a simulated household agent, which interprets and performs tasks issued by a user in written language. The household agent moves in a virtual world, which includes, among other things, a fridge, a dishwasher, a TV, a sofa (for the "Master"), a mop and a bucket (for cleaning the floor), and random dirt.

---

<sup>1</sup> [www.isi.edu/isd/LOOM/LOOM-HOME.html](http://www.isi.edu/isd/LOOM/LOOM-HOME.html)

<sup>2</sup> [www.ghg.net/clips/CLIPS.html](http://www.ghg.net/clips/CLIPS.html)

The agent operates in auto-mode, as long as there is no user input, and will do regular chores like cleaning the floor. The agent has some kind of "self-awareness" and returns to its charging station whenever it notices that its batteries are low in power. The user (Master) can issue command statements in natural language to the agent, for example, to switch on the TV or get a drink or food item, like "Bring me a glass of milk".

In case of a verbal command input, the household agent analyzes the natural language instruction and builds a case-frame-like representation. Based on the essential components of this frame-representation, in this example the action "bring", with destination "Master" (me), and the object "glass of milk", the agent finds respective actions in the conceptual action hierarchy and develops suitable plans, if necessary. Using the action and the object hierarchies, the agent determines a higher level action "bring", and fills respective features of this action with information from the case frame. In this example, the object of the *bring*-action is instantiated as a glass containing milk, e.g. a yet unspecified instance "glass#1" of type "glass", which is a sub-class of the concept "container" with feature "contents=milk". This information can be inferred from the object hierarchy. The *bring*-action involves two sub-actions: one is to get the respective object (i.e. a glass of milk) and the other one is to deliver it (i.e. to the Master). In order to fulfill the preconditions of the generic *bring*-action, the agent<sup>3</sup> must have the object: "have(agent,object)" or, with unified variables: "have(agent,glass#1)", and the agent must be at the destination location, i.e. the Master's sofa, to deliver it. In order to fulfill the first condition, the agent performs a *get*-action, and thus starts a planning process to get a glass of milk, which involves going to the dishwasher, picking a glass from the dishwasher, going to the fridge and filling the glass with milk. The precondition of the *deliver*-action states that the agent has to be at the Master's location. Thus, the agent plans now a route to the location of the Master (i.e. the sofa) and moves there. The actions referred to at this level, e.g. get a glass from the dishwasher, are translated into executable actions for the household agent. The execution of the move itself, based on a planned route involving obstacle avoidance etc., is similarly assumed to be executable by the agent.

In auto-mode, i.e. if the agent has no instruction to do a specific task, it performs standard tasks, e.g. if the floor is dirty (recognized based on its perceptions), it will perform a "clean(floor)" action, involving as above planning and action instantiation processes, e.g. to get a mop and a bucket.

### 5.3 Speech Controlled Toy Car

A speech interface for a remote controlled toy car has been developed using the Dragon™ speech recognition software. The speech interface for the remote controlled toy car allows the user to issue spoken commands like 'forward', 'right', or 'north', 'east' etc. instead of using the manual remote control. The verbal speech commands are hooked to menu items in a window which is again connected to a parallel output port. Signals arriving at this port are transmitted through a custom-made electronic switching circuit to the remote control. This system is supposed to be extended with a more complex natural language interface and command interpreter, which allows the processing of complex command sentences like "Make a slight turn right and then stop."

---

<sup>3</sup> Since we have a single household agent in our system, the term "agent" is used like constant, referring to an individual concept or instance.

## 6 Conclusion

In this paper we presented a framework for action descriptions and its connection to natural language interfaces for artificial agents. The core point of this approach is the use of a generic action/object hierarchy, which allows interpretation of natural language command sentences, issued by a human user, as well as planning and reasoning processes on the conceptual level, and connects to the level of agent executable actions. The linguistic analysis is guided by a case frame representation, which provides a connection to actions (and objects) represented on the conceptual level. Planning processes can be implemented using typical precondition and effect descriptions of actions in the conceptual hierarchy. The level of primitive actions (leaf nodes of this conceptual hierarchy) connects to the agents' executable actions. The level of primitive actions can thus be adapted to different types of physical agents with varying action sets.

Further work will include the determination of a suitable general action ontology, possibly based on FrameNet [9], which an enrichment of preconditions and effects. Other topics to be pursued related to communicating through natural language with mobile physical agents include the representation and use of contextual information and spatial constraints, in order to resolve ambiguities and to detect and block impossible or unreasonable actions.

## Acknowledgements

The prototype systems described here have been developed and implemented by the author in co-operation with graduate and undergraduate students. Thanks to all of them for their participation and enthusiasm. This work has been partially supported by NSERC.

## References

1. J. F. Allen, B. W. Miller, E. K. Ringger, and T. Sikorski, A Robust System for Natural Spoken Dialogue, *Proc. Annual Meeting of the Association for Computational Linguistics (ACL'96)*, pp. 62-70, 1996.
2. J. F. Allen, et al. The TRAINS project: A Case Study in Defining a Conversational Planning Agent, *J. of Experimental and Theoretical AI*, 7, 1995, pp.7-48.
3. A. Artale and E. Franconi, A Temporal Description Logic for Reasoning about Actions and Plans, *Journal of Artificial Intelligence Research*, 9, pp. 463-506, 1998.
4. F. Baader, D. Calvanese, D. McGuinness, D. Nardi, P. Patel-Schneider (eds.), *The Description Logic Handbook*, Cambridge University Press, 2003.
5. R. J. Brachman and J. G. Schmolze, An Overview of the KL-ONE Knowledge Representation System, *Cognitive Science*, 9(2), pp. 171-216, 1985.
6. B. Coyne and R. Sproat, WordsEye: An Automatic Text-to-Scene Conversion System, *Siggraph*, 2001. See also: Semantic Light, [www.semanticlight.com](http://www.semanticlight.com)
7. P. T. Devanbu and D. J. Litman, Taxonomic Plan Reasoning, *Artificial Intelligence*, 84, pp. 1-35, 1996.
8. B. Di Eugenio, An Action Representation Formalism to Interpret Natural Language Instructions, *Computational Intelligence*, 14, pp. 89-133, 1998.



9. C. F. Baker, C. J. Fillmore and J. B. Lowe: The Berkeley FrameNet Project. *COLING-ACL*, Montreal, Canada, 1998.
10. C. Fillmore. 1968. *The case for case*. In Emmon Bach and Robert Harms, editors, *Universals in Linguistic Theory*, pages 1--90. Holt, Rhinehart and Winston, New York.
11. D. Jurafsky and J. H. Martin, *Speech and Language Processing*, Prentice-Hall, 2000.
12. C. Kemke, Speech and Language Interfaces for Agent Systems, *Proc. IEEE/WIC/ACM International Conference on Intelligent Agent Technology*, pp.565-566, Beijing, China, September 2004.
13. C. Kemke, A Formal Approach to Describing Action Concepts in Taxonomical Knowledge Bases. In: N. Zhong, Z.W. Ras, S. Tsumoto, E. Suzuku (Eds.), *Foundations of Intelligent Systems*, Lecture Notes in Artificial Intelligence, Vol. 2871, Springer, 2003, pp. 657-662.
14. C. Kemke, What Do You Know about Mail? Knowledge Representation in the SINIX Consultant. *Artificial Intelligence Review*, 14:253-275, 2000.
15. C. Kemke, *About the Ontology of Actions*. Technical Report MCCS -01-328, Computing Research Laboratory, New Mexico State University, 2001.
16. C. Kemke, Die Darstellung von Aktionen in Vererbungshierarchien (Representation of Actions in Inheritance Hierarchies). In: Hoepfner (ed.), *GWAI-88, Proceedings of the German Workshop on Artificial Intelligence*, Springer, 1988.
17. C. Kemke, Representation of Domain Knowledge in an Intelligent Help System, *Proc. of the Second IFP Conference on Human-Computer Interaction INTER-ACT'87*. pp. 215-200, Stuttgart, FRG, 1987.
18. M. Montemerlo, J. Pineau, N. Roy, S. Thrun, and V. Verma. Experiences with a Mobile Robotic Guide for the Elderly. *AAAI National Conference on Artificial Intelligence*, Edmonton, Canada, 2002.
19. J. McCarthy. *Formalizing Common Sense: Papers by John McCarthy*. Ablex Publishing, 1990.
20. P. F. Patel-Schneider, B. Owsnicki-Klewe, A. Kobsa, N. Guarino, R. MacGregor, W. S. Mark, D. L. McGuinness, B. Nebel, A. Schmiedel, and J. Yen. Term Subsumption Languages in Knowledge Representation. *AI Magazine*, 11(2): 16-23, 1990.
21. A. Stent, J. Dowding, J. M. Gawron, E. Owen Bratt and R. Moore. The CommandTalk Spoken Dialogue System. *Proc. 37<sup>th</sup> Annual Meeting of the ACL*, pp. 183-190, University of Maryland, College Park, MD, 1999.
22. M. C. Torrance. *Natural Communication with Robots*, S.M. Thesis submitted to MIT Department of Electrical Engineering and Computer Science, January 28, 1994.
23. D. Traum, L. K. Schubert, M. Poesio, N. Martin, M. Light, C.H. Hwang, P. Heeman, G. Ferguson, J. F. Allen. Knowledge Representation in the TRAINS-93 Conversation System. *Int. J. of Expert Systems 9(1), Special Issue on Knowledge Representation and Inference for Natural Language Processing*, pp. 173-223, 1996.
24. S. Thrun, M. Beetz, M. Benniswitz, W. Burgard, A.B. Cremers, F. Dellaert, D. Fox, D. Haehnel, C. Rosenberg, N. Roy, J. Schulte, and D. Schulz. Probabilistic Algorithms and the Interactive Museum Tour-Guide Robot Minerva. *Intern. Journal of Robotics Research*, 19(11):972-999, 2000.
25. W. Wahlster. *VERBMOBIL: Erkennung, Analyse, Transfer, Generierung und Synthese von Spontansprache*. Report, DFKI GmbH, Juni 1997.
26. E. Walker. *An Integrated Planning Algorithm for Abstraction and Decomposition Hierarchies of Actions*. Honours Project, Dept. of Computer Science, University of Manitoba, 2004.
27. R. Weida and D. Litman. Subsumption and Recognition of Heterogeneous Constraint Networks. *Proceedings of CAIA-94*, pp. 381-388, 1994.

# An Extended BDI Agent with Policies and Contracts

Bei-shui Liao<sup>1</sup>, Hua-xin Huang<sup>1</sup>, and Ji Gao<sup>2</sup>

<sup>1</sup> Research Center of Language and Cognition, Zhejiang University,  
310028 Hangzhou, China  
baiseliao@zju.edu.cn, rw211@zju.edu.cn

<sup>2</sup> Institute of Artificial Intelligence, Zhejiang University,  
310027 Hangzhou, China  
gaoji@mail.hz.zj.cn

**Abstract.** In order to enable the intelligent agents to be aware of the dynamic business requirements and strategies, and cooperate with other agents in a stable and explicit way, a policy and contract extended BDI logic (called  $BGI_{PDC}$  logic) has been proposed, by integrating contracts and policies into traditional BDI model. On the basis of  $BGI_{PDC}$  logic, this paper proposes a model of agent architecture as a concrete realization of it, called PDC-agent. PDC-agent is an extension of traditional BDI agent, by adding a policy engine, a contract engine and a goal maintenance component into agent's interpreter. Besides, PDC-agent has two characteristics. First, the operation of PDC-agent is based on an event-driven mechanism. Various events drive the components of the interpreter. Second, the representation of PDC-agent is on the basis of ontology.

## 1 Introduction

Currently, agent has been a promising candidate for the autonomic management of service integration under the context of virtual organization [1-3]. As autonomic elements [4], agents should not only be autonomous, but also be aware of the dynamic business requirements and be able to cooperate with other agents according to contracts [5]. In [5], we have proposed a model of multi-agent system based on policies and contracts and its fundamental part, policy and contract extended agent model, called  $BGI_{PDC}$ . In  $BGI_{PDC}$  logic, agent's internal desires, and the obligations that are brought about by policies and contracts (called PObligations and CObligations respectively) are accommodated into the extended BDI logic. These various kinds of motivations are taken as goals of PDC-agent after conflict treatment, so that the agent based on  $BGI_{PDC}$  logic (called PDC-agent [6]) is capable of acting an autonomic element in autonomous service integration systems.

Based on  $BGI_{PDC}$  logic and other work presented in [7-9], this paper proposes a model of agent architecture (PDC-agent) from the perspective of realization. The structure of this paper is organized as follows. In section 2, an ontology language for PDC-agent representation is introduced. In section 3, the structure of PDC-agent is proposed. In section 4 and 5, the representation of PDC-agent and the working principles of it are formulated respectively. And finally, in section 6, conclusions are drawn.

## 2 Ontology Language for PDC-Agent Representation

In PDC-agent model, the representation of policies, contracts and agent's internal elements (beliefs, desires, PObligations, CObligations, goals, intentions, actions, etc.), is based on domain concepts, which are represented by domain conceptualization language (DCL) [10]. DCL is defined in the form of EBNF as follows.

```

Ontology ::= `Ontology(' OntologyID [VersionDeclaration]
                [OntologyCitation] [SynonymousConcepts]
                [SynonymousProperties] [PropertyDefini-
                tions]
                [ConceptDefinitions] [TypeDefinitions] `)'
ConceptDefinitions ::= `{ `Concept('ConceptName
                [ `Super:' {SuperClassName}]
                {SlotName ':' {AspectName AspectContent `,'}
                ;'} [ `Constraint: 'ConditionExpression] `)' }
AspectName ::= `val' | `type' | `mode' | `number' | `derive' |
                `restriction' | `unit' | `inverse' | `superslot'

```

In this paper, predicate is represented as a Concept Instance Pattern (CIP), which is defined in the form of EBNF as follows.

```

InstancePattern ::= `( @ `ConceptName
                {SlotName ':' SlotValue} `)'
SlotValue ::= [ '?' ] String
ParameterValue ::= [ '?' ] String

```

For example, as to a concept, *compose service*:

```

Concept(O_CompServ ServID:type String ServProperties:
        type C_ServProperties),

```

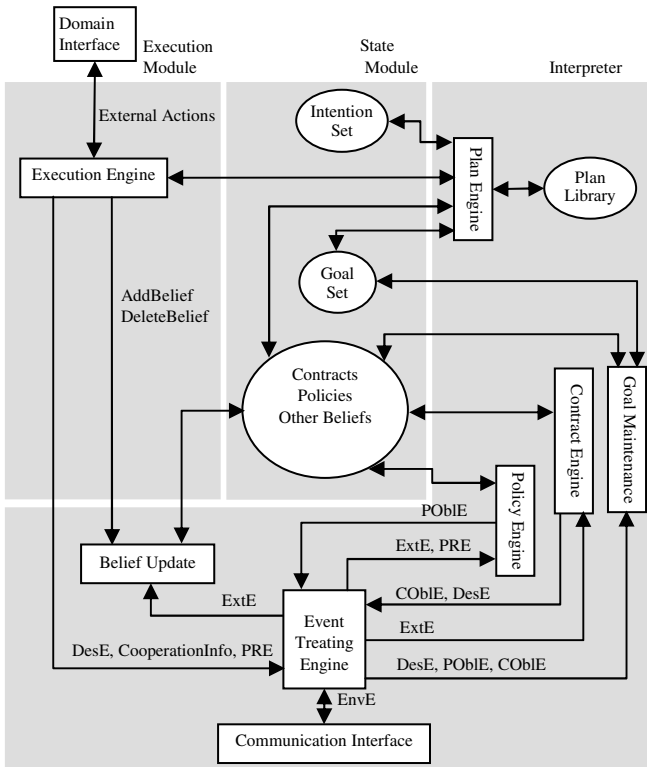
its instance pattern (`@O_CompServ ServID: ?x ServProperties: ?y`) is equal to the predicate represented in the conditional way “(CompServ ?x ?y)”.

## 3 The Structure of PDC-Agent

As shown in Fig.1, a PDC-agent is composed of an *Interpreter*, a *State Module* and an *Execution Module*. Among them, the Interpreter is defined as a 7-tuple: (Event Treating Engine, Belief Update, Contract Engine, Policy Engine, Goal Maintenance, Plan Engine, Plan Library), in which

- the *Event Treating Engine* is in charge of receiving and manipulating various events, including external events (ExtE), agent desires events (DesE), cooperation information events (CooperationInfo), policy reference events (PRE), and the obligations events from the Contract Engine and the Policy Engine respectively; the detailed definitions of events are in section 4.4;
- the *Belief Update* component updates beliefs according to ExtE, belief-updating actions (AddBelief, DeleteBelief), and current beliefs;

- the *Contract Engine* triggers contract clauses, including fulfillment clauses, authorization clauses and violation punishment clauses, according to current valid contracts, environment events and beliefs, and generates CObligation events (COblE) or desire events (DesE);
- the *Policy Engine* evaluates policies according to ExtE, PRE and current beliefs, and generates PObligation events (POblE);
- the *Goal Maintenance* component receives DesE, POblE, and COblE, treats with possible conflicts among the three kinds of motivations (desires, PObligations, and CObligations), and produces goals;
- and the *Plan Engine* conducts means-end reasoning according to current goals and available plans in the *Plan Library*.



**Fig. 1.** The structure of PDC-agent

The State Module is composed of Intention Set, Goal Set and beliefs, in which beliefs are divided into policies, contracts, and other beliefs. Finally, the *Execution Module* contains an *Execution Engine*, some external actions and internal actions. Among them, internal actions are used to update beliefs and send desire events (DesE), policy reference events (PRE) and cooperation information events (CooperationInfo) to the Event Treating Engine, while external actions are used to invoke local domain services to realize some application logics.

The operation of PDC-agent is based on an event-driven mechanism. Various events drive the components of the Interpreter, which continually execute the following cycle:

- the Event Treating Engine observes the world and sends the ExtE to the Belief Update component, Policy Engine and Contract Engine respectively;
- the Belief Update component updates beliefs according to current beliefs and the ExtE;
- according to current beliefs and related events, the Policy Engine evaluates policies and generates PObligation events (PObIE), while the Contract Engine evaluates contracts and generates CObligation Events (CObIE) and internal desire events (DesE); in turn, PObIE, CObIE, and DesE are respectively sent to the Event Treating Engine;
- after the Event Treating Engine receives the PObIE, CObIE and DesE, it sends them to the Goal Maintenance component respectively;
- the Goal Maintenance component receives the PObIE, CObIE and DesE, treats with conflicts among them, and sends them to the Goal Set (in the State Module) respectively;
- the Plan Engine circularly checks the Goal Set, takes out a goal (if the Goal Set is not empty), and generates a plan instance according to current beliefs and the available plans in the Plan Library, then pushes the plan instance onto an existing or new intention stack, according to whether or not the event is a subgoal;
- the Plan Engine selects an intention stack, takes the topmost plan instance, and drive the Execution Engine executing the next step of this plan instance; if the step is a domain action, the Execution Engine directly performs it; if it is a goal or an action that should be performed by external agents, the Execution Engine encapsulates it as a cooperation information (CooperationInfo) and sends it to the Event Treating Engine; in turn, the latter sends the Cooperation-Info to the Communication Interface; otherwise, if it is a subgoal, the status of the current plan is set to *suspended*, and the subgoal is capsulated as a new desire event (DesE), which is sent to the Event Treating Engine.

In this way, the emergence of external events will bring about PObligations, CObligations, or internal desires. These motivations drive the instantiation of plans. When a plan starts executing, its subgoals are taken as new desire events (DesE) and sent to the Event Treating Engine. Then, the DesE will be sent to the Goal Maintenance component and become new goals, which will bring about further instantiation of plans.

## 4 Representation of PDC-Agent

In this section, the various elements of PDC-agent (beliefs, desires, PObligations, CObligations, goals and actions, etc.) will be respectively presented based on the DCL and CIP.

#### 4.1 Beliefs, Desires, PObligations, CObligations, Goals and Actions

**Beliefs.** According to the  $BGI_{PDC}$  logic [5], PDC-agent's beliefs are composed of contracts, policies, facts about the world, and internal states, etc. A belief  $BEL(i, \phi)$  is represented in terms of DCL as follows.

```
Concept(C_Belief AgentID: type String, BelContent: type
  C_BelContent)
```

The type of  $BelContent$  is a concept, which is embodied by its sub-concepts: contracts, policies, facts, and states. First, a contract [5] consists of several clauses, which can be divided into three categories: fulfillment clauses, authorization clauses and punishment clauses. Formally, a contract clause is defined as  $Obl_N(i, j, \phi|\psi)$ ,  $Pms_N(i, j, \phi|\psi)$ , or  $Prh_N(i, j, \phi|\psi)$ , in which  $N$  denotes the authority for arbitration when fulfilling violation punishment;  $i$  denotes obligation fulfilling agent, while  $j$  denotes its counterpart;  $\phi$  and  $\psi$  are the objective and the condition of a contract respectively. On the other hand, a policy is defined as a rule that reflects high-level business strategies and requirements. Formally, an enforceable policy is represented as  $Obl(i, j, \phi|\psi)$ ,  $Pms(i, j, \phi|\psi)$ , or  $Prh(i, j, \phi|\psi)$ . The detailed policy representation and refinement is referred to [5, 8]. The representation of contract clause and policy is based on DCL and CIP. For instance,  $Obl_N(i, j, \phi|\psi)$  is equal to the following concept:

```
Concept(C_ContractClause Super: C_BelContent
  Au_AgentID: type String, OF_AgentID: type String,
  OC_AgentID: type String, OblContent: type C_Objective
  Condition: type C_ConditionExpr)
```

In this definition, concept  $C\_ContractClause$  has a super concept  $C\_BelContent$ , and five slot names:  $Au\_AgentID$ ,  $OF\_AgentID$ ,  $OC\_AgentID$ ,  $OblContent$  and  $Condition$ , denoting  $N$ ,  $i$ ,  $j$ ,  $\phi$  and  $\psi$  in the  $Obl_N(i, j, \phi|\psi)$  respectively.

Second, the representation of facts and states is also based on DCL and CIP.

**Desires, PObligations and CObligations.** PDC-agent has three kinds of motivations: desires  $DES(i, \phi)$ , PObligations  $PObl(i, j, \phi)$ , and CObligations  $CObl_N(i, j, \phi)$ . The representation of them is based on DCL and CIP:

```
Concept(C_Desire AgentID type: String,
  DesContent: type C_DesContent)

Concept(C_PObligation OF_AgentID:type String,
  OC_AgentID: type String, OblContent: type
  C_Objective)

Concept(C_CObligation Au_AgentID: type String,
  OF_AgentID:type String, OC_AgentID: type String,
  OblContent: type C_Objective)
```

In these definitions, the type of obligation content ( $OblContent$ ) is a concept  $C\_Objective$ , which is defined in the form of EBNF as follows:

```
C_Objective ::= "!" | "?" InstancePattern
```

in which, the notation "!" and "?" denote "achieve" and "query" respectively.

**Goals.** Goals of PDC-agent are produced from the above three kinds of motivations after conflict treatment (a rule-based mechanism is adopted for conflict treatment, as presented in paper [5]). The representation of goals is the same as that of C\_Objective.

**Actions.** Actions of PDC-agent are divided into two categories: internal actions and external actions. The former includes *add beliefs* (AddBelief), *delete beliefs* (DeleteBelief), *Add desire events* (AddDes), *execute test objectives* (ExTestObj), *add policy reference events* (AddPRE) and *add cooperation information* (AddCooperationInfo); the latter is domain specific, and can be defined as a function call.

## 4.2 Plans

Plans of PDC-agent are defined in the form of EBNF as follows.

```
AgentPlan ::= `AgentPlan('PlanID Trigger
                    [PreCondition], PlanBody, [MaintCondition],
                    PostProcessing)`

Trigger ::= `Trigger('GoalEvent`)'
PreCondition ::= `PreCondition('{ConditionExpression}`)'
MaintCondition ::= `MaintCondition('
                    {ConditionExpression}`)'
PostProcessing ::= `PostProcessing('{AddBelief
                    |DeleteBelief }`)'`
```

In this definition, Trigger defines the triggering condition of a plan (In this paper, it can only be a goal event); Precondition defines the pre-conditions of a plan, which are determined by current beliefs; MaintCondition defines the maintenance conditions that must be true for the plan to continue executing; and Postprocessing defines actions that are performed if the plan fails.

Finally, plan bodies (PlanBody) are expressed as a tree, in which nodes denote states, and branches denote internal actions or external actions. Successfully executing of a plan means traversing the tree from the root (start state) to any leaf node (end state). A plan body can be defined in the form of EBNF as follows.

```
PlanBody ::= EndState | { `(` Planbody `(' State
                    PlanBranch `)` `)` }

EndState ::= InstancePattern
State ::= InstancePattern
PlanBranch ::= ExternalAction | InternalAction
ExternalAction ::= FunctionCall
InternalAction ::= AddBelief | DeleteBelief | AddDes |
                    ExTestObj | AddCooperationInfo | AddPRE
AddBelief ::= `AddBelief('Belief `)`'
DeleteBelief ::= `DeleteBelief('Belief `)`'
AddDes ::= `AddDes('Des `)`'
ExTestObj ::= `ExTestObj('TestObjective `)`'
AddCooperationInfo ::= `AddCooperationInfo('
                    CooperationInfoE `)`'
```

```

AddPRE := `AddPRE('PRE')`
Des ::= AchieveObjective
AchieveObjective ::= `!` InstancePattern
TestObjective ::= `?' InstancePattern

```

### 4.3 Plan Instances

When the Goal Set is not empty, the Plan Engine will take out a goal, and match it with the *Trigger* of the available plans in the Plan Library. If there are some plans whose *Triggers* are satisfied, then select one of these plans and instantiate it. Plan instances are defined as follows.

```

PlanInstance ::= `PlanInstance('PlanID InstanceID
                               PlanEnvironment CurrentState
                               Choices IntendedBranch Status `)'
PlanEnvironment ::= `PlanEnvironment('{('Parameter ` , '
                                       Value`)'}`
CurrentState ::= `CurrentState('State | EndState`)'
Choices ::= `Choices('{Branch}`)'
IntendedBranch ::= `IntendedBranch('[Branch]`)' Status
                ::= `Status('`Active' | `Suspended`)'

```

In this definition, *PlanEnvironment* is the environment of the plan, representing a set of bindings that have been generated in the course of executing the plan; *CurrentState* is the current state reach in the plan; *Choices* are the set of branches that can be traversed from this state; *IntendedBranch* is the branch it is attempting to traverse; and *Status* is the status of the plan (either “active” or “suspended”).

### 4.4 Intentions and Events

**Intentions.** The intention of PDC-agent is operationalized as a sequence of plan instances. As to a specific goal, the Plan Engine creates an intention, which contains the generated plan instance. If this plan includes a sub-goal which can’t be performed directly, then it is regarded as a new desire and further become a new goal to which the agent responds with another plan, the new plan is concatenated to the intention. In this way, the plan at the top of the intention stack is the plan that will be executed first in any intention.

**Events.** In PDC-agent, events are divided as internal events (IntE) and external events (ExtE). The former includes desire events (DesE), PObligation events (POble), CObligation events (COble), policy reference events (PRE), and cooperation information events (CooperationInfoE); the latter includes policy events (PolicyE), contract events (ContractE) and environment events (EnvironmentE). Formally, events are defined in the form of EBNF as follows.

```

Event ::= `Event('EventType EventContent [DesContext]`)'
EventType ::= IntE | ExtE
EventContent ::= ConceptInstance
DesContext ::= `DesContext('PlanInstanceID
                          PlanEnvironment FailedPlanInstSet `)'

```



```

IntE ::= 'DesE' | 'POblE' | 'COblE' | 'PRE'
      | 'CooperationInfoE'
ExtE ::= 'PolicyE' | 'ContractE' | 'EnvironmentE'
PolicyE ::= 'AddPolicyE' | 'DelPolicyE' | 'UpdPolicyE'
ContractE ::= 'NewContractE' | 'UpdContractE'
          | 'ConcelContractE'

```

In this definition, an event mainly includes two parts: event type and event content. The latter is application specific, represented as a concept instance. In addition, as for a desire event which occurs when the branch of an executing plan at the top of an intention is an achieve goal that cannot be achieved immediately, it includes a context (DesContext) in addition to the above two parts. The DesContext contains a plan instance identifier (PlanInstanceID) that causes the desire event, a plan environment (defined in the previous section), and a set of plan instances that may already have failed and may not be retried (FailedPlanInstSet). Besides, policy reference events (PRE) are produced during the execution of a plan instance, which provide a means for agent's requesting the guidance of policies.

## 5 Working Principles of PDC-Agent

The working principles of PDC-agent will be formulated from the following four aspects, including event manipulation, motivation producing, goal maintenance, intention generation and execution.

**Event manipulation.** The Event Engine is in charge of the manipulation of various events, and works according to following algorithm:

- Step1. wait for new events;
- Step2. when receive a new event, manipulate it according to the different cases:
  - case1: if the event is an environment event (ExtE), send it to the Policy Engine, Contract Engine, and the Belief Update component respectively;
  - case2: if the event is a policy reference event, send it to the Policy Engine;
  - case3: if the event is a motivation event (PObl, CObl, or DesE), send it to the Goal Maintenance component;
  - case4: if the event is a cooperation event, send it to the communication interface;
- Step3. return to Step1.

**Motivation producing.** In PDC-agent, motivations are generated from the Policy Engine, Contract Engine, and the execution of plan instance.

First, the Policy Engine evaluates a policy according to the ExtE, PRE, and the current state of the policy. Within a life cycle, a policy has four states: *active*, *evaluating*, *waiting*, and *executing*. The state transition of a policy is shown in Fig.2. When a policy is deployed [7], its initial state is “*active*”. Then, once an ExtE or a PRE happens, the Policy Engine evaluates all “*active*” policies according to a specific arrangement strategy (sequently or concurrently); the states of the arranged policies will turn to “*evaluating*”. If an “*evaluating*” policy passes the evaluation, i.e., its condition

(Trigger and/or Constraint [7]) is satisfied, its state will turn to “waiting”, and a PObligation will be produced. When the state of a policy is “waiting”, the PObligation event related to it will be sent to the Event Treating Engine. Then, its state turns to “executing”. If the PObligation is rejected during the goal maintenance stage (due to the conflicts between the PObligation and other motivations), or is successfully finished, then its state turns to “active”, so that it can be evaluated at the next turn.

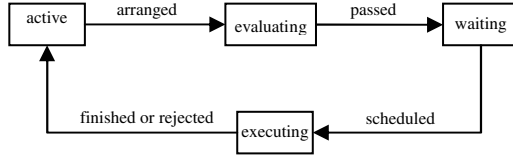


Fig. 2. The state transition of a policy

Second, the structure of Contract Engine is shown in Fig.3. The contract clauses defined in [5] are divided into three categories, including fulfillment clauses, punishment clauses, and authorization clauses. Correspondingly, the Policy Engine is composed of three components: Fulfillment Clause Evaluation (FCE), Punishment Clause Evaluation (PCE), and Authorization Clause Evaluation (ACE).

Upon receiving an ExtE, the FCE, PCE and ACE evaluate the contract clauses respectively. In the fulfillment clauses  $Obl_N(i, j, \phi \psi)$  and  $Obl_N(j, i, \phi \psi)$ , when  $\psi$  becomes true, the FCE of agent  $i$  generates a desire (acquiring a service) and an obligation (providing a service) respectively. Similarly, if the condition ( $\psi$ ) of a punishment clause is true, the PCE will generate a desire to punish the counterpart; and if the condition ( $\psi$ ) of an authorization clause is true, the ACE will produce an authorization.

After the generation of motivations, the Contract Engine encapsulates the motivations as desire events (DesE) or a CObligation Events (COblE) and sends them to the Event Treating Engine.

Third, when the Plan Engine puts a plan instance into execution, if this plan includes a sub-goal which can't be performed directly, a new desire will be generated and sent to the Event Treating Engine.

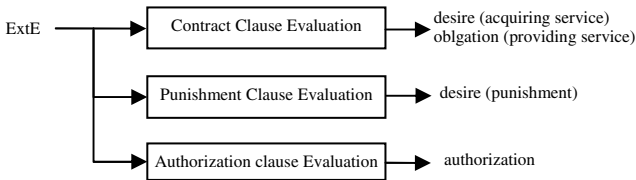


Fig. 3. The structure of the Contract Engine

**Goal maintenance.** Goal Maintenance component receives motivation events (DesE, POblE, and COblE) from Event Treating Engine, treats with possible conflicts among them, and produces goals, according to current beliefs and intentions. The conflict treatment is based on the following rule.

**Rule1.** When a PDC-agent has to make a choice for a goal from three different motivations (desire, PObligation, and CObligation), if two motivations among them have the same goal, then it should choose this goal; else, if these motivations are mutual exclusive, then it should choose an alternative preferred by its CObligations first, one preferred by its PObligations second, and one preferred by its own desires third.

**Intention generation and execution.** Intention Engine matches a goal with the *Trigger* of the available plans. If there is no matching plan, the goal is identified as “failed”; else, one of them is selected and manipulated in a way according to the type of the goal:

- when the goal is a sub-goal of an existing plan instance, the Intention Engine instantiates a new plan with the plan environment related to the plan instance, then checks this new plan instance, if it belongs to the set of failed plans, give up it, or else put the new plan instance on to an existing intention stack;
- when the goal event is a PObligation event, the Intention Engine first checks the type of the policy’s *Trigger*; if it is an ExtE, then instantiate a new plan and put the new plan instance onto a new intention stack, or else put the new plan instance onto an existing intention stack;
- when the goal event is a CObligation event or a desire event brought about by a contract, then instantiate a new plan, and put the new plan instance onto a new intention stack.

After intentions are created, the Plan Engine selects one of intention and puts it into execution. Then, the Plan Engine locates current state of the plan instance and determines the *Choices*. According to the type of the branch, the plan instance is executed in the following ways respectively:

- if the branch is an external action, initiate the Execution Engine, which invokes the external action;
- if the branch is to add a belief or delete a belief, initiate the Execution Engine, which invokes the internal action to update the beliefs;
- if the branch is to add a desire event, a cooperation information event, or a policy reference event, send it to the Event Treating Engine;
- if the branch is to execute a query sub-goal, perform it according to the current beliefs.

## 6 Conclusions

In this paper, a model of agent architecture based on  $BGI_{PDC}$  logic (PDC-agent) was proposed. PDC-agent is an extension of the traditional BDI agent [9], and is the concrete realization of  $BGI_{PDC}$  logic. Compared to the traditional BDI agent, the characteristics of PDC-agent are as follows. First, the components of its beliefs not only include general beliefs similar to those of traditional agent, but also involve contracts and policies; second, a Policy Engine, a Contact Engine, and a Goal Maintenance component are added to the Interpreter for obligation generation and motivational conflict treatment, so that the agent’s behaviors are determined by the internal

motivation, i.e., its own desires that reflect the designed objectives and the variations of environments, and the external motivations, i.e., the obligations brought about by policies and contracts respectively; third, components of the Interpreter are driven by various events, in which the cooperation information events (CooperationInfo) are manipulated and sent to other agents by the Event Treating Engine and the Communication Interface, so that the PDC-agent is able to cooperate with other agents; finally, the representation of PDC-agent is based on CIP, in which the *slot names* are defined explicitly as part of a concept, so compared to traditional predicates in which parameters are not defined, the CIP has explicit, machine understandable semantics.

## Acknowledgements

We gratefully acknowledge the support of the National Grand Fundamental Research 973 Program of China under Grant No.2003CB317000, and the 985 Project of Zhejiang University, China.

## References

1. J. Patel, W. T. L. Teacy, N. R. Jennings, et al. Agent-based virtual organisations for the Grid. *Int J. Multiagent and Grid Systems*, 1 (4), 2005.
2. I. Foster, N. R. Jennings and C. Kesselman. Brain meets brawn: Why Grid and agents need each other. In *Proceedings of the 3rd International Conference on Autonomous Agents and Multi-Agent Systems*, New York, USA, 8-15, 2004.
3. Jeff O. Kephart and David M. Chess. The Vision of Autonomic Computing[J]. *IEEE Computer*, 2003;36(1): 41~50.
4. Bei-shui Liao, Ji Gao, Jun Hu, Jiujun Chen. A Model of Agent-Enabling Autonomic Grid Service System. In *proceedings of GCC 2004, LNCS 3251*, pp.839-842, 2004.
5. Bei-shui Liao, Ji Gao. A Model of Multi-agent System Based on Policies and Contracts. In *proceedings of CEEMAS 2005, LNAI3690*, pp.62-71, 2005.
6. Bei-shui Liao, Ji Gao. Dynamic Self-Organizing System Supported by PDC-Agents. *Journal of Computer-Aided Design & Computer Graphics*, 2006, 18(2).
7. Bei-shui Liao, Ji Gao, Jun Hu, Jiujun Chen. Ontology-Based Conceptual Modeling of Policy-Driven Control Framework: Oriented to Multi-agent System for Web Services Management. In *proceedings of AWCC2004, LNCS 3309*, pp. 346-356, 2004.
8. Bei-shui Liao, Ji Gao. An Automatic Policy Refinement Mechanism for Policy-driven Grid Service Systems. In *proceedings of GCC2005, LNCS 3795*, pp. 166-171, 2005.
9. M. P. Georgeff and A. L. Lansky. Reactive reasoning and planning[A]. In *proceedings of AAAI-87*, 1987. 677~682.
10. Ji Gao, Cheng-xiang Yuan, Jing Wang. SASA5: A Method System for Supporting Agent Social Activities. *Chinese Journal of Computers*, 2005, 28(5): 1-11.

# Towards a Customized Methodology to Develop Multi-Agent Systems

Xiao Xue<sup>2</sup>, Xingquan Liu<sup>1,\*</sup>, and Rong Li<sup>3</sup>

<sup>1</sup> Zhejiang Forestry University, Lin'an, Zhejiang, P.R. China  
Tel.: +86-571-63732757; Fax: +86-571-63740809  
bolowliu@hotmail.com

<sup>2</sup> Institute of Automation, Chinese Academy of Sciences, BeiJing, P.R. China  
jzxuexiao@126.com

<sup>3</sup> Lucent Technologies China Co. Ltd, BeiJing, P.R. China  
rongl@lucent.com

**Abstract.** Despite a great deal of research, there still exists a number of challenges before making agent-based computing a widely accepted paradigm in software engineering practice. In order to realize an engineering change in agent oriented software engineering, it's necessary to turn agent oriented software abstractions into practical tools for facing the complexity of modern application areas. The paper presents a customizable development architecture for multi-agent systems, which can empower the developer to assemble a methodology tailored to the given project by putting appropriate meta models together, much like developers building applications from third party off-the-shelf components. To exemplify its feasibility and effectivity, the construction of C4I system is presented as a case study.

## 1 Introduction

Agents and multi-agent systems (MASs) have recently emerged as a powerful technology to face the complexity of a variety of today's ICT (information and communication technology) scenarios, such as distributed system, web service and so on. What's more, the emergent general understanding is that MASs, more than an effective technology, represent indeed a novel general-purpose paradigm for software development [1]: agent-based computing can promote designing and developing applications in terms of agents, situated in an environment, and that can flexibly achieve their goals by interacting with one another in terms of high-level protocols and languages.

In the last few years, there has been a great deal of research related to the identification and definition of suitable models and techniques to support the development of complex software systems in terms of MASs [2], such as new metaphors, formal modelling approaches, development methodologies and modelling techniques, specifically suited to the agent-oriented paradigm.

However, despite the great deal of research in the area, agent oriented software engineering (AOSE) is a relative young area, and there are, as yet, not standard methodologies, development tools, or software architectures. There still exists a number of challenges before making agent-based computing a widely accepted paradigm in software

---

\* Corresponding author.

engineering practice[3]. To face the challenges, it's the key to turn agent-oriented software abstractions into practical tools for facing the complexity of modern application areas.

In the paper, we propose a customizable development architecture for MASs to handle the challenge, which can cover the whole development lifecycle: from agent oriented analysis to software implementation. Based on the architecture, developers can combine different meta models originated from various AO methods to form the best suited development process for particular application domain, much like developers building applications from third party off-the-shelf components.

The remainder of the paper is organized as follows. Section 2 introduces current research work about AOSE. Section 3 represents the core of the paper: it introduces the customizable development architecture for MASs, and the key concepts and supporting mechanism. Section 4 discusses how to apply the development architecture in the construction of C4I system. Some conclusions are presented in section 5.

## 2 The Current Situation of AOSE

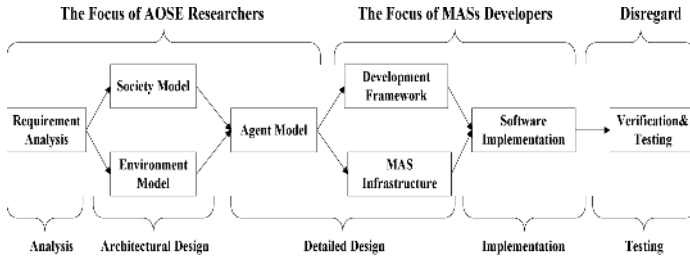
### 2.1 The Defects of Current AO Methodologies

At the moment, people have made a lot of attempts at AOSE, such as the building of MASs, and on the definition of suitable models, methodologies and tools for these systems. The AOSE uses the notion of an autonomous agent as its fundamental modeling abstractions to map from items in the real-world to objects in the computational domain, which is useful both for the effective implementation of abstract problem solutions and for the management of software complexity. Agent-oriented abstractions, as a new, powerful approach, would carry several advantages [1]:

- autonomy of application components, enforces a stronger notion of encapsulation (i.e., encapsulation of control rather than of data and algorithms), which reduces the complexity of managing systems with a high and dynamically varying number of components;
- taking into account situatedness explicitly, and modelling environmental resources and active computational entities in a differentiated way, provides for a better separation of concerns which, in turn, helps reduce complexity;
- dealing with dynamic and high-level interactions enables to address in a more flexible and structured way the intrinsic dynamics and uncertainties of modern distributed scenarios.

Nevertheless, the research of AOSE is still in its early stages, and a lot of challenges need to be handled before AOSE can deliver its promises, becoming a widely accepted and a practically usable paradigm for the development of complex software systems.

As shown in figure 1, different AO researchers focus on different aspects. In order to pursue generality, some AO methods take agent as a modeling unit and only emphasize agents' external characteristics and their relationships between each other, e.g. the Gaia methodology[9]. However, the other AO methods are bound to some type of concrete agent architecture to facilitate software implementation, e.g. the MaSE methodology[10]. Some research works (e.g. [4]) provide comparison studies between different AO methodologies, showing that each AO method has its own weaknesses and strengths.



**Fig. 1.** The MAS development process

However, software engineers that are going to spend money and man-months in the development of a complex software system will hardly be convinced to shift to a new paradigm (and pay the overhead involved in such a process), simply because it is conceptually elegant. Instead, they will require some evidence of the fact that this will help them save money and resources.

Against this background, it's often infeasible or non-efficient to use a single kind of AO method to solve all the problems in the construction of MASs. Depending on the concrete goal and requirement, different kinds of AO methodologies and strategies might be necessary during each development phase to optimally control the development process. Thus, the advantages of different AO methods can be taken advantage of and their drawbacks can be overcome. Furthermore, such exploitation speeds up development, avoids reinventing the wheel, and enables sufficient time to be devoted to the value added by the multi-agent paradigm. In the end, developers can obtain a suited development process to meet the demand for particular application.

## 2.2 The Defects of Current Development Process

At present, the emphasis of AOSE is still focused on the advantages that agents could bring to software systems only, but not on the advantages that agents could bring in the process of developing software systems. A large number of AOSE methodologies describing how the process of building a MAS should/could be organized have been proposed in the literature [5]. What characterizes most of the methodologies proposed so far is that they assume a very traditional waterfall model (from analysis to design, implementation, and maintenance) for organizing the process of building a MAS.

It is a matter of fact that such a standard model is rarely applied in the real world of industrial software development. In a majority of the cases, software is developed following a non-structured process: analysis, design, and implementation, often collapse into the frenetic work of a bunch of technicians and programmers, directly interacting with clients, and striving to deliver the work on time. In the mainstream community of software engineering, such a situation is getting properly attributed via the definition of novel software process models, specifically conceived to give some flavor of "engineering" to such chaotic and frenetic processes (e.g., agile and extreme software process models).

Currently, no AO methodology can cover the whole development process and can completely solve how to achieve a concrete implementation based on the model

derivation from the system design. Different MASs varies a lot in the demand for development process based on system size and application domain[3]:

- *The micro level* - A limited number of agents have to be defined to interact toward the achievement of a specific application goal. Developers may typically preserve and enforce a strict control and understanding over each and every component of the system.
- *The macro level* - Huge software systems with a very large number of interacting agents, possibly deep in dynamic and uncontrollable operational environments, e.g. swarm systems and the Grid. The collective behavior of the system is what matters.
- *The meso level* - Engineers will have to face the two basic questions of (i) what will be the impact on my system of it being deployed in an open and complex scenario and (ii) what will be the impact on the global system of the deployment of our own systems.

In order to be suited for different application domains, the development process for MASs needs to shift from standard to non-standard and extreme processes. The development process should be customizable, i.e. it's not only suited for large scale systems, similar to heavyweight RUP(Rational united process), but also suited for small scale systems, just like agile programming. It will play a key role to realize a engineering change in AOSE.

### 2.3 The Existing Solutions to the Problems

In order to improve the situation, some research[6,7] have been made in the field of creating and reusing modular methodologies. The existing fruits are summarized as below:

- **Variety** - Future large-scale software development projects will require engineering support for a diverse range of software quality attributes, such as privacy and openness. It is not feasible to create one monolithic methodology to support all possible quality attributes.
- **Combination** - There is no silver bullet when pursuing a tailoring methodology, so the best approach is combining existing work according to the given domain or situation. The AOSE methodologies are expected to be created and reused in a modular way, which enables developers to build custom project-specific methodologies from AOSE features.
- **Customization** - An AOSE feature encapsulates software engineering techniques, models, supporting CASE tools and development knowledge. Based on the concept of the AOSE features, engineers can perform one or more development activities, such as analysis, and addresses one or more quality attributes, such as privacy. In the end, applications are built from reusable off-the-shelf components.

Despite of all the progress, there is still a long way to go in the field. Our final goal is not to pursue some isolate and maybe optimal meta model, but a suited development process for MASs. Unfortunately, the key point is often disregarded. Only a coarse-grain development route(analysis - design - implementation - testing - release) is



provided to integrate all AOSE features. Engineers are confused about how to assemble a group of meta models to form an optimal development process.

Some researchers propose that the automation of case tool will facilitate the process, especially for those developers without background in agent research. However, the author admits that “such an approach may not be useful for certain applications, such as ones with unclear requirement”[6]. Unfortunately, MASs are often more complex than traditional system. It’s difficult to identify and define all requirements without the revision and the trade-off between different requirements iteratively.

If users are only suited to the automatic process passively and don’t know why to do this, they will be restricted a lot and can’t improve the process positively to meet particular demand. Therefore, an explicit development architecture for MASs is needed to assemble all those meta models. Based on the architecture, user can understand how to fit each meta model in the development process where appropriate to form a best suited development process.

### 3 The Customized Development Architecture for MASs

#### 3.1 The Customized Development Architecture

According to the above analysis, our vision is to empower the developer to assemble a methodology tailored to the given project by putting appropriate AOSE meta models together, much like developers building applications from third party off-the-shelf components. The methodology becomes a living artifact of the project, and changes according to system requirements.

One approach is to have an all-embracing methodology in the way that Rational Software promotes the Rational Unified Process. This is problematic for two reasons: (a) it needs to be constantly updated and possibly refactored to take into account new developments, and (b) it promotes lock-in to a particular supplier of a methodology, which is often undesirable.

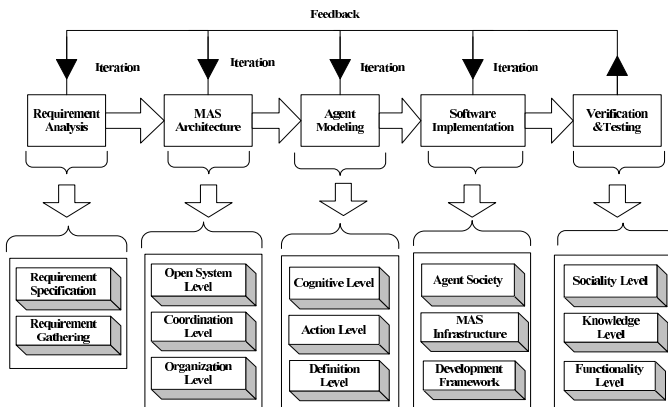


Fig. 2. The MAS development architecture

In figure 2, an alternative and modular development architecture for MASs is proposed as an improvement. The development architecture is used to produce an ordered sequence of steps, an identifiable set of models, and an indication of the interrelationships between the models, showing how and when to exploit which models and abstractions in the development of a MAS.

The whole development process is divided into five phases: requirement analysis, MAS architecture, agent modeling, software implementation and verification, which covers the whole development lifecycle: from requirement analysis to software implementation. The process is iterative. The analyst or designer are allowed to move between steps and phases freely such that with each successive pass, additional detail is added and eventually, a complete and consistent system design is produced.

At the same time, each phase is categorized into a layered model structure further. Meta models can be created to handle all kinds of quality attributes and be filled into the layers where appropriate. Together, the layers enable the configuration of a new AO approach that can be customized for specific application by combining various AO methodologies.

Based on the modular development process, each new project has its own customized process, built up from components of other methodologies. If one part of the system requires a stringent view with respect to privacy, then a methodology that incorporates privacy well can be adopted for that part. If team structures need to be modeled in another part, choose a methodology that supports teams for that part.

The benefits of a modular approach are clear. Instead of creating incompatible techniques, models and CASE tools for each methodology, modular and reusable solutions can be created once, and shared within different methodologies. It represents significant savings in development cost and learning cost.

In order to guarantee the alternative development architecture to be successful and effective, three problems needs to be cared :

1. How to fashion meta models into the development process?
2. How to abstract suited meta model which is isolated from specific methodologies?
3. How to ensure that models originated from different AO methods can keep semantic consistency?

### 3.2 The Meta-modal

Reusing existing models, techniques and tools when possible, rather than constructing new model from ground up, can reduce the learning curve for model users, and reduce the risk of replacing proven methods with untested new practices. Here, the meta model serves as the basic building block for filling in the development architecture, as well as providing the basis that enable developers to engineer such systems in a robust, reliable and repeatable fashion. It's a key issue to create and abstract the meta model from those merged AO methods in the application of the architecture.

After determining the scope and the purpose of the existing methodology, we need to partition the existing methodology into multiple meta models and determine the dependencies between the new models and any existing models. Based on the development tasks and quality goals, the meta model is created by isolating general-purpose common

features from existing AOSE methodologies. The remaining parts of the methodologies are then componentized into special purpose "value-adding" features.

The meta-model does not solve application-specific engineering problems, but provides a clean high-level structure where engineering issues can be grouped, classified and accommodated. Therefore a meta-model should promote scalability of systems to accommodate the application complexity. The criteria for the abstraction of meta model is given below:

- **Independence:** A meta-model should clearly separate agent knowledge from low-level functionalities. The knowledge and the functionalities can then be developed, reused, and maintained separately, at a much lower cost. A meta-model adhering to this principle also facilitates engineering of quality attributes at the knowledge level.
- **Modularization:** Knowledge should be developed and maintained in modular units with high cohesion and low coupling. The modular approach localizes potential faults and enables easy sharing and reuse of knowledge.
- **Runtime characteristics:** A meta-model should not place arbitrary constraints on the level of agent characteristics such as level of intelligence, autonomy, proactiveness and reactivity. At the same time, a meta-model should allow the nature and level of quality attributes to be changed at runtime.
- **Simplicity:** A meta-model should not be unnecessarily complex, so the resulting methodologies, tools and languages don't inherit the complexity and unnecessarily burden from the development of systems.
- **Ease of Learning:** For better acceptance of AOSE, a meta-model should be easy to learn and understand by industry practitioners. In addition to being simple, the meta model should also be similar to existing approaches such as OOSE, so the user can leverage their existing knowledge for easy transition.

According to those criteria, a meta model may include a comprehensive list of potential elements, similar to the format of defining design pattern[8]:

**1. Model Name:** It is a handle we can use to describe a design problem, its solutions and consequence in a word or two.

**2. Model Description:** describes when to apply the metal model. It might describe the development tasks the model supports, the quality attributes the model delivers, the required properties of the system and the application domain. Sometimes, the problem will include a list of conditions that must be met before it makes sense to apply the model.

**3. Solution:** describes the elements that make up the design; their relationships, responsibilities and collaborations; the semantics of AOSE concepts used in them. The solution doesn't describe a particular concrete design or implementation, because a meta model is like a template that can be applied in many different situations. Instead, the pattern provides an abstract description of a design problem and how a general arrangement of elements solves it.

**4. Support:** defines CASE tools which support the development tasks; programming support which includes programming frameworks, programming languages, reusable

components, and reusable knowledge bases; development knowledge that helps the development tasks to be achieved, including design patterns, sample design and implementation, analysis of the application domain, best practices, tutorials and learnt lessons.

**5. Consequence:** the results and trade-offs of applying the meta model. They are critical for evaluating design alternatives and for understanding the costs and benefits of applying the meta model.

### 3.3 The Layered Modeling

The principle of “mechanism separating from strategy” is popular in software development. The meta model is the underlying mechanism of AOSE, which give users the freedom of customizing development process. However, for most of engineers, especially those absent in agent technology, the freedom may become a heavy burden. Thus, there is a thirsty for a set of rules and strategies to guide how to assemble those meta models to form a suited development process. The idea of “layered modeling” is presented as the strategy to meet the demand.

In the layered structure, the bottom level is the basis of the up level. Each successive level introduces greater implementation to satisfy the original requirements statement. By means of those layers, the increasing details of the phase to be constructed are developed step by step. In the end, we can obtain a sufficiently detailed design that can be implemented directly from a depiction of system.

The layered modeling viewpoints can be defined as a technique for suppressing unnecessary details according to different abstraction levels, providing us with an appropriate separation of concerns regarding system analysis and design. The division in abstraction levels shows us that we naturally should target the modeling task from different perspectives.

The principles of identifying layers is mainly decided by the goal and implementation mechanism in each phase. The details are given in the following context:

- **Requirement analysis:** gather, identify and define requirements according to practical application, including functional and non-functional. Then, the requirement statements will be mapped to a MAS, i.e. the requirements are specified in the view of agent, role and organization.
- **MAS architecture:** define how multi agents construct the whole system and realize the required functions. Firstly, the whole system is conceived in terms of an organized society of individuals in which each agent plays specific roles and interact with other agents according to protocols determined by the roles of the involved agents. Furthermore, the component agents can be explicitly designed to cooperatively achieve a given goal, the coordination behavior derives from the interaction among the constitute agents.

As a more general case, agents might not be co-designed to share a common goal, and have been possibly developed by different people to achieve different objectives. Moreover, the composition of the system can dynamically vary as agents enter and leave the system.

- **Agent modeling:** make clear three aspects of each agent to pave the way for the final software implementation: what to think of? how to think? how to react outside? Aimed at the three aspects, the whole phase is divided into three layers: definition, action and cognitive.
- **Software implementation:** To obtain usable software products, the process is divided into three steps: the agent society layer can be reused, meaning that it is independent of the MAS infrastructure layer, and finally the framework layer depends on the implementation platform of our choice. In MAS infrastructure layer, if we apply JADE or other AO framework, we must use other models that comply with the products provided by this specific framework. Regarding the framework layer, we can select the suited development tools and programming language. If we use Java, we may use UML Class Diagrams for this last abstraction level in agent society layer.
- **Verification and Testing:** ensure that system is stable, reliable, effective and meet all levels of requirements: functional level, knowledge level, sociality level.

A layer/model table can be constructed to facilitate filling meta models from various AO methodologies into the development process. The layer/model table is created by identifying and listing all meta models in columns, mapping against development layers in rows, and filling quality tag and reasons in the supported cells. An example is shown below:

Phase and Layers AO Meta Models	Phase Name		
	Layer1 Name	Layer2 Name	Layer3 Name
Model1 Name	✓ (Reason)		
Model2 Name		✓ (Reason)	

**Fig. 3.** The layer/model table

The table shows that which models are used for each modeling aspect, i.e. the layer1, layer2 and layer3. These three aspects are, in general, targeted in some system analysis and design phase. The developers can then group meta models together by development phases and layers. Using the layer/model table and various analysis on the project, the most appropriate combination can be determined.

### 3.4 The Potential Problems

The attempt to emerge different approaches, in addition to increasing the complexity of the modeling, is also likely to introduce some potential problems. In the area of AOSE, most of AO methodologies will build on the basis of the existing work, few of that begin from scratch. As a result, besides their own characteristics, there are many similarities between them in concepts and functions.

The difference or similarity result in the dangerous mismatch or overlap between the abstraction level adopted and the conceptual level at which application problems have to be solved. The combination of different AO methodologies is a complex and trade-off process. To ensure the success of the process, the semantic consistency and the function overlap need to be paid attention to. Otherwise, these problems will lead the whole development process to a mess.

Some researchers[7] propose that some kind of transformation model can be applied to go from one model to the other model in a consistent way. A mapping table is used as tools to depict the transformation rules, meaning that a concept in one method must be mapped to its counterpart in the other method. In this way, the analysts and designers are provided with some guidelines on how to convert one to the other. To some extent, the problems can be avoided.

Furthermore, a generic meta-model is expected to presented as a more general solution, which allows the agent organization and various quality attributes to be expressed independent of agent architectures and knowledge representation formats. By conforming to the meta-model, the resulting development process is expected to exhibit basic semantic consistency.

At the same time, a meta-model only encapsulates the essential functional unit. If a selected model uses more detailed agent concepts such as belief and intention from the BDI architecture, then the model must include the precise definition of additional agent concepts used from the selected AO methodology. When a meta model is created, the developer should ensure that the model does not overlap with neighboring models in function.

Currently, there are still many potential problems which seek to be solved from a theoretical viewpoint. We expect common sense to guide the development of meta model and a practical engineering perspective is taken to enable the development of a large set of meta models that are carefully kept consistent with each other.

## **4 The Construction of the C4I System**

### **4.1 The Project Description**

The C4I(command, control, communication, computer and Information) system is the core of the whole naval warship, which is used as information process(including collection, transformation, process and transmission), fighting support and weapon control. The C4I system needs to ensure that different kinds of weapons can cooperate effectively and the integrated performance can be improved to a high degree. Apart from the(problem-solving) functionality, the C4I system must also satisfy properties such as reliability, fault-tolerance, maintainability, transparency, scalability etc..

In order to achieve the goal, three main problems need to be solved during the construction of C4I system: firstly, how to harmonize a number of components which may have been developed by teams having worked separately on different portions of the system; secondly, how to adopt new technology to deal with multiple, heterogeneous and even dynamic application environments; thirdly, how to integrate many different technologies, languages, paradigms and legacy systems together in an effective and fruitful

way. It's difficult for traditional software engineering methodologies to deal with those problems conveniently.

In the C4I system, different components can be viewed as autonomous, situated, and social agents and be devoted to controlling and directing different stages of the physical process: information collection - information process - fighting support - command - weapon control. Therefore, we attempt to apply agent oriented technology as an original and more effective way to solve highly-complex problems calling for system intelligence.

On the basis of the proposed architecture, we represent a new approach by merging several existing AOSE methodologies: Gaia[9], MaSE[10], and ZEUS [11]. The new approach is aimed at distributed problem solving systems in which the component agents are explicitly designed to cooperatively achieve a given goal. The class of system is closed; all agents are known a priori, they are supposed to be innately cooperative to each other and, therefore, they can trust one another during interactions.

In a research project for C4I system development, the new approach is adopted to solve the problems in the construction of C4I system on naval warship. The C4I system is an example of the distributed problem solving system. Through applying the new approach, a large-scale C4I system is decomposed successfully and reconstructed conveniently. In figure 4, we provide a sketch map of the whole system. The system owns the advantages of distributed system, e.g. resource sharing, scalable and flexible. What's more, many legacy systems can be encapsulated and taken advantage of, which makes system more robust and reliable.

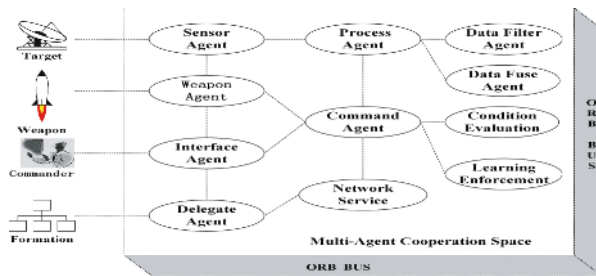


Fig. 4. The agent oriented C4I system architecture

It makes novel use object-oriented ontology for specifying the content of agent messages, and employs the OMG's common object request broker architecture(CORBA) to provide the foundation for agent communication.

## 5 Conclusions

The definition of agent-specific methodologies is definitely one of the most explored topics in AOSE. However, there is still a long way to go in the field because of the defects in current AO methodologies and development process. In order to realize an

engineering change in AOSE, we face the challenge of how to turn AO software abstractions into practical tools for facing the complexity of modern application areas.

The paper presents a development architecture for MASs to empower the developer to assemble a methodology tailored to the given project by putting appropriate AOSE meta models together. Based on the architecture, developer can customize his own development process to meet the demand of different domains. To exemplify its concepts and show its feasibility and effectiveness, it is introduced to merges several representative AO methodologies. The new approach is applied in the construction of C4I system on warship and the experimental result is satisfactory.

## References

1. N.R.Jennings.: An agent-based approach for building complex software system. *Commun, ACM*, vol.44, no.4, 35-41.
2. M.Gervais, J.Gomez and G.Weiss.: A survey on agent-oriented software engineering researches. *Methodologies and Software Engineering for Agent Systems, Kluwer: NewYork(NY)* (2004).
3. F.Zambonelli, A.Omicini.: Challenges and research directions in agent-oriented software engineering. *Autonomous Agents and Multi-Agent System* (2004) 253-283.
4. Khanh Hoa Dam and Michael Winikoff.: Comparing Agent-Oriented Methodologies. In Proc. of 5nd International Workshop on Agent Oriented Software Engineering(2003).
5. O.Shehory, A.Sturm.: Evaluation of modeling techniques for agent-based systems. In Proceedings of the 5th International Conference on Autonomous Agents, ACM Press:Montreal(2001).
6. T.Juan, L.Sterling, M.Martelli, V.Mascardi.: Customizing AOSE Methodologies by Reusing AOSE Features. In Proc. 2nd Int. Conference on Autonomous Agents and Multi-Agent Systems, Melbourne Australia (July, 2003).
7. R.S.S.Guizzardi, V.Dignum, A.Perini, G.Wagner.: Towards an Integrated Methodology to Develop KM Solutions with the Support of Agents. In Proceedings of the International Conference on Integration of Knowledge Intensive Multi-Agent Systems, Waltham, Massachusetts (2005).
8. Erich Gamma, Ralph Johnson, Richard Helm, John M. Vlissides.: *Design Patterns: Elements of Reusable Object-Oriented Software*, Addison-Wesley (October,1994).
9. F. Zambonelli, N. R. Jennings, and M. Wooldridge.: Developing multiagent systems: The gaia methodology. *ACM Trans. Softw. Eng. Methodol*(2003) 12(3):317-370.
10. S.A.Deloach, M.F.Wood, C.H.Sparkman.: Multiagent System Engineering. *Software Engineering and Knowledge Engineering*(2001) 11(3):231-258.
11. Jaron Collins and Divine Ndumu.: ZEUS methodology documentation. <http://www.labs.bt.com/projects/agents/index.htm>



# A Systematic Methodology for Adaptive Systems in Open Environments

Li-ming Wang<sup>1</sup> and Ya-chong Li<sup>2</sup>

School of Information & Engineering,  
Zhengzhou University,  
450052 Zhengzhou, China  
<sup>1</sup> ielmwang@zzu.edu.cn  
<sup>2</sup> liyachong@163.com

**Abstract.** Gaia is one of the first agent-oriented software engineering methodologies, which explicitly takes social concepts into account. Yet Gaia would neither suffice to adequately develop adaptive system in open environments nor describe role relation and hierarchy of organizational rule. On the other hand, a variety of models employed by different methodologies limit the progress of MAS, so it is necessary to develop a unified framework which includes the models adopted by different methodologies. FRAG is proposed as a systematic methodology, which is the extension of role models of Gaia on the relation among function, role, and agent, for developing adaptive systems in open environments. Having compared several methodologies, the ORRA process is proposed as a universal process in role-based methodology starting from the purpose of different models adopted in MAS methodology. The Conference Management Case study is introduced to exemplify ORRA's process and to show the use and effectiveness of FRAG's models in the development of MAS.

**Keywords:** Adaptive Systems, Open Environments, Organization Rule, Role Space.

## 1 Introduction

Juan [1] points out that intelligent adaptive system capable of handling open environments have significant commercial value for industry. Haiping Xu [2] gives open MAS two means: open agent society and open role space.

Gaia [3] is one of the first agent-oriented software engineering methodologies, which explicitly takes social concepts into account. After extending Gaia1 [4], Gaia2 can be utilized in open agent society environments. Luca [5] points out that Gaia is suitable for a design-for-change perspective because of its clear separation from the analysis and the architectural design phases. Unfortunately, Gaia is not entirely suitable for developing adaptive system in open environments. Although Gaia supports "design-for change" and "open agent society", Gaia does not have explicit models to support development of "design-for change" and "open role space" System. Furthermore, although organizational rule [6] and role is two important abstracts of Gaia, Gaia would not suffice to adequately describe hierarchy of organizational rule

and role relation (inheritance, aggregation, association and incompatibility [2]). Gaia doesn't concern the organization rules between partial entities and have no models to support aggregative relation and inheritable relation.

On the other hand, while some study was focus on meta-model adopted by different methodology [7, 8], few work is concerned the purpose of why such meta-model was proposed, only Alexander [9] et al. want to give some principles about a universal process.

The use of FRAG, which is based on the improved role model of Gaia, is advocated as a methodology for developing adaptive systems in open environments. Furthermore, the ORRA (Organization specification, identifying Role and Relation, assigning Agent role) process is proposed as a universal process for Role-base MAS methodology.

The remainder of this article is organized as follows: Section 2 details the basic definitions and FRAG models. Section 3 focuses on ORRA process. It also compares difference between models of several methodologies used in ORRA process. Section 4 describes the use and effectiveness of FRAG methodology and ORRA process using a representative example: Conference Management. Section 5 concludes.

## 2 Basic Definitions and FRAG Models

### 2.1 Basic Definitions

#### Definition 1. Role

A role is a tuple  $R = \langle N, \Sigma_A \rangle$  where  $N$  is the name of the role,  $\Sigma_A$  the set of attributes {Permission, Responsibility, Activity, Protocol}.

#### Definition 2. The context of role

The context of role is a tuple

$$\Theta(R) = \langle \bigcup_{\omega \in \Omega} Sup(\omega), \bigcup_{\omega \in \Omega} Sub(\omega), \Omega \rangle$$

Where  $Sup(\omega)$  is the Super-Role of node  $R$  and  $Sub(\omega)$  is the Sub-Role of node  $R$  while  $\omega \in \Omega$ ,  $\Omega$  is the set of relation type {inheritance, aggregation, association, incompatibility}. When  $\omega \in \Omega$ ,  $\Theta_\omega(R)$  means the relation  $\omega$  context of  $R$ .

#### Definition 3. Role Space

Role Space is a tuple

$$\Gamma = \langle V, \bigcup_{v \in V} \Theta(v) \rangle$$

where  $V$  is the set of role,  $\Theta(V)$  is the context of the role.

#### Definition 4. Dimension

"Multiple inheritance means that a class has multiple superclass" [10]. Multiple inheritance is a characteristic of roles. Dimension means a type of inheritance relation, Dimension set is denoted by  $\Phi$ .

**Definition 5.** The context of R with dimension  $\phi$

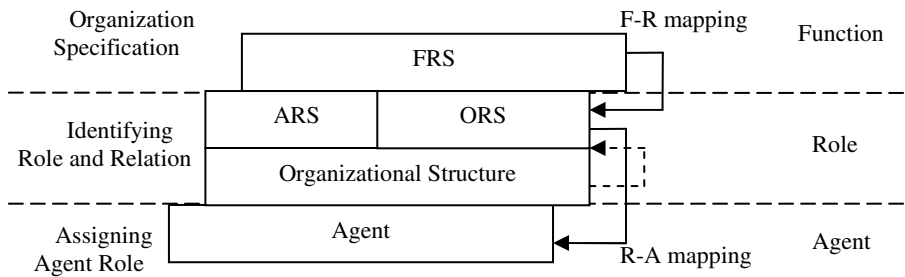
The context of R with dimension is a tuple

$$\Theta_{\omega(R)-\phi} = \langle \bigcup_{\omega \in \Omega, \phi \in \Phi} Sup(\omega, \phi), \bigcup_{\omega \in \Omega, \phi \in \Phi} Sub(\omega, \phi), \Omega, \Phi \rangle$$

Where  $Sup(\omega, \phi)$  is Super-Role of R where  $\omega \in \Omega$  and  $\phi \in \Phi$

## 2.2 FRAG Models

FRAG (Function-Role-Agent of Gaia) focuses on the relationship among **Function**, **Role** and **Agent**. It classifies roles into three categories: FR, OR, and AR according to the different use and effectiveness in analysis and design of organization. **Context model** is adopted to describe the relation between roles. Accordingly there are three kinds of Role Space: **FRS**, **ORS** and **ARS**. ORRA process is detailed in section3. Fig 1 shows the models of FRAG used in ORRA process.



Virtual arrow means only if organizational structures have been determined; FRAG can complete the role model and interaction model.

**Fig. 1.** The models of FRAG used in ORRA process

**Function:** Function describes what the organization is and what the organization wants to do.

**Role:** Role is an abstract concept that represents the static structure of organization.

**Agent:** Agent is an activity entity that represents the dynamic running of organization.

**F-R mapping:** F-R mapping describes how to design an organization. F-R mapping not only reflects the function requirement of organization also implicitly involves non-function requirement of organization. FRAG uses aggregation of roles to implement F-R mapping.

**R-A mapping:** R-A mapping describes how an organization can run better. R-A mapping is used to assign role to agent dynamically. FRAG accomplishes R-A mapping task through agent Take/Release role.

### 2.2.1 FRS Model

Function is accomplished by means of activities and protocols of FR (Function Role). **FR** is the abstract correspondence to organizational function and the **FRS** (Function Role Space) Model is composed of FR and  $\Theta(\text{FR})$ . FRS Model decomposes function of organization into more specific function. Decomposing function is a process of roles refinery and the last outcomes of decomposing function is **AtR** (Atomic Role). Every FR has its Super-role and Sub-role. There is a **part-of** relationship between FR and its Super-role, so is between FR and its Sub-role.

The necessary condition of FR is: No more than two AtR are evolved to accomplish one function.

If two roles have the same function but other attributes are different (i.e. same activity, but different permission), then they are regarded as different AtR, which is a difference from Objective Normalization used by ARL.

#### Definition 6. FRS

$$\text{FRS} = \langle \Sigma_{\text{FR}}, \Theta_{\text{ag}}(\text{FR}) \rangle$$

Where  $\Sigma_{\text{FR}}$  is the set of all the FR,  $\Theta_{\text{ag}}(\text{FR})$  is the aggregation context of FR, i.e.  $\Theta_{\text{ag}}(\text{FR}) = \langle \bigcup_{\omega=\text{ag}} \text{Sup}(\omega), \bigcup_{\omega=\text{ag}} \text{Sub}(\omega) \rangle$ .

### 2.2.2 ORS Model

**OR** (Organization Role) is the role existing in the static structure of organization, and **ORS** (Organization Role Space) model is composed of OR and  $\Theta(\text{OR})$ . AtR is not suitable for direct participation of organization interaction as OR for the sake of convenience or effectiveness. Wood [11] gives some reasons about why goals are combined into a single role. F-R mapping AtR to OR through aggregation reflects the effect of non-function requirement. The relation between OR and AtR is **part-of** relation.

#### Definition 7. ORS

$$\text{ORS} = \langle \Sigma_{\text{OR}}, \Theta_{\text{ag}}(\text{OR}) \rangle$$

Where  $\Sigma_{\text{OR}}$  is the set of all of OR,  $\Theta_{\text{ag}}(\text{OR})$  is the aggregation context of OR, i.e.  $\Theta_{\text{ag}}(\text{OR}) = \langle \bigcup_{\omega=\text{ag}} \text{Sup}(\omega), \bigcup_{\omega=\text{ag}} \text{Sub}(\omega) \rangle$ .

### 2.2.3 ARS Model

**AR** (Abstract Role) means a kind of OR, and **ARS** (Abstract Role Space) model is composed of AR and  $\Theta(\text{AR})$ . The abstractivity of role is one of six characteristics of role defined by Bent Bruun Kristensen et al [12] which is extensively used in OO. The relation between AR and  $\Theta(\text{AR})$  is **is-a** relation. So AR is used in two ways: 1 implements the hierarchy of organizational rules; 2 identify common attributives and reuse these attributives. Sometimes AR is regarded as Role Type [13].

**Definition 8. ARS**

$$ARS = \langle \Sigma_{AR}, \Theta_{in}(OR) - \phi \rangle$$

Where is the set of all the AR,  $\Theta_{in}(OR) - \phi$  is the inheritance context of OR with  $\phi$

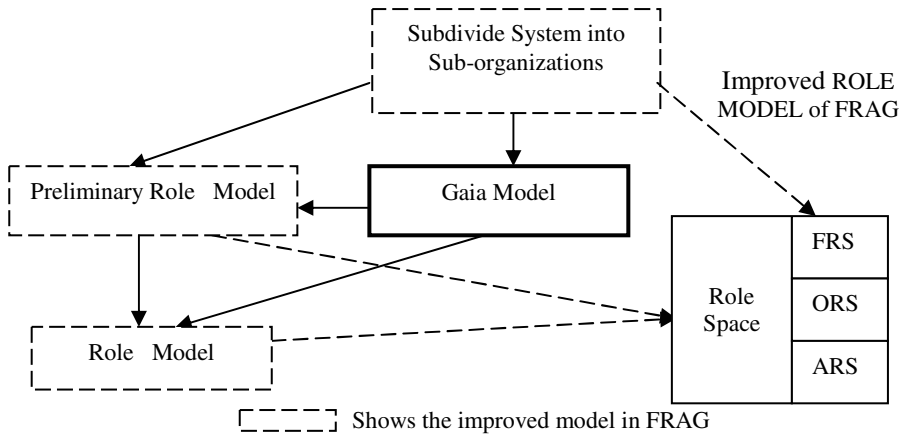
i.e.  $\Theta_{in}(OR) - \phi = \langle \bigcup_{\omega=In, \phi \in \Phi} Sup(\omega, \phi), \bigcup_{\omega=In, \phi \in \Phi} Sub(\omega, \phi) \rangle .$

**2.2.4 RS Model**

**RS** (Role Space) model is composed of FRS model, ORS model and ARS model. Unlike the definition of role space by Haiping Xu [2], FRAG does not distinguish Conceptual Role and Concrete Role. With the introduction of the role space model, MAS can dynamically add or delete roles according to the needs while the system is running. Only if organizational structures have been determined, FRAG can complete the role model and interaction model, as does Gaia. All the roles of role space are but preliminary roles.

**2.2.5 Improvement of FRAG**

FRAG reserves the entire model (i.e. interaction model, organizational rules model) and process except role model. Fig 2 shows the improvement of FRAG to Gaia.



**Fig. 2.** The improvement of FRAG to Gaia

The improvement of FRAG to Gaia is as below:

1. Instead of the first model (divides system into sub-organizations) in Gaia, FRS is the model adopted to describe organization function. FRAG gives a way to decompose function and condition of terminate compare with Gaia;
2. FR generates OR by aggregation, and OR participates organization interaction. From FR to OR is F-A mapping that explicitly separating the organization structure from Organization function;

3. AR as participant in organizational rules realizes the hierarchy of organizational rules, and it also implements reusing of roles.

When  $\Gamma$  is determined by organization structure, FRAG can assign role dynamically to agent through R-A mapping.

### 3 ORRA Process

Different methodologies adopt different terminologies and provide different abstractions and few standardization works have been done yet, limiting the impact of agent design on the industrial world [7]. Both Carole Bernon [7] and Xinjun Mao [8] have presented the unified Meta-model. However, it is difficult to create a unified meta-model that can be utilized by most methodologies. Instead of building a unified model of MAS methodology, we advocate the universal ORRA process of MAS methodology starting from the purpose of different models adopted by MAS methodology. ORRA process includes three steps:

**Organization Specification:** A preliminary analysis of organization, this is a preliminary description of organization. No matter what models is adopted by methodology to describe organization goal, Organization Specification is an absolutely necessary process.

**Identifying Role and Relation:** This step identify role and relation between roles based on the analysis outcomes of first step.

**Assigning Agent role:** There are two perspectives about relation between role and agent:

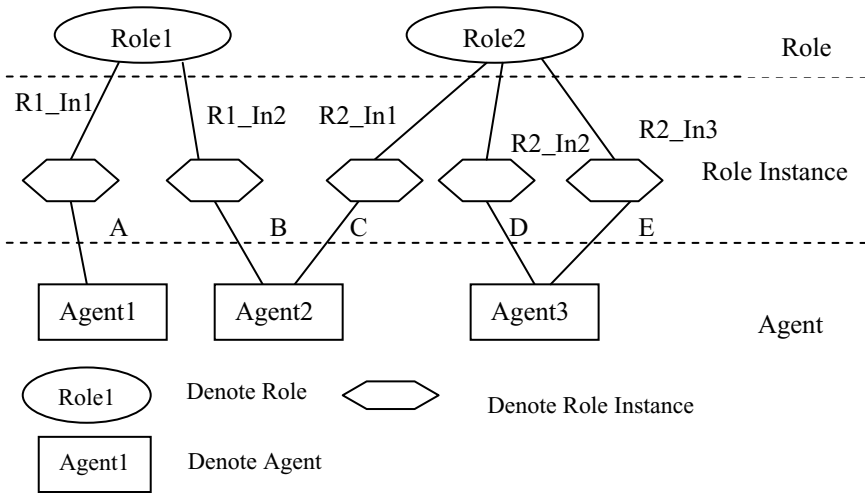
- a. Agent Type is independent to role, Assigning agent role through judging whether a role type is suitable for any agent type (e.g. ARL).
- b. There is a correspondence between roles and agent classed, role as an abstract concept used in analysis and design, and agent as an active software entity playing a set of agent roles (e.g. Gaia). Whatever any perspective is taken by methodology, Role must be assigned to agent dynamically.

Although different role-related models adopted by different methodology, they can belong to a universal process. Table1 shows the models and abstracts used in the ORRA process by Gaia, MaSE [11], Roadmap [14], ARL [15] and FRAG.

After having determined the roles in the organization, FRAG could dynamically assign the role to Agent which is completely independent from role. A role could own multiple role instances and there are three relationships between Agent and role. In general one Agent could hold one role instance. But it is also possible that one Agent hold two role instances of the same role or one Agent hold two role instances of different roles. The rule what role an agent can hold is decided by the incompatibility relationship of roles. In Fig3 the relationship between role and Agent is shown.

**Table 1.** The models and abstracts used in the ORRA process by different methodologies

Methodology	Organization Specification	Identifying Role and Relation	Assigning Agent Role
Gaia	Sub-organizations	Role, Interaction	Role-Agent Class
MaSE	Goal Hierarchy	Role	Role-Agent Class
Roadmap	USE Case	Role	Role/Agent Hierarchy
ARL	Objective Normalization	SuperRole/SubRole	Agent-role locking
FRAG	FRS	F-R mapping	R-A mapping



**Fig. 3.** The relations between role and agent

Line A denotes one to one relation between R1\_IN1 and Agent1 which means one Agent holds one role instance. Line B and C denote that Agent2 holds role instance R1\_IN2 of Role1 and R2\_IN1 of Role2 which means one Agent could hold multiple different role instance of different roles. Line D and E denote that Agent3 simultaneously holds instance R2\_IN2 and R2\_IN3 of Role2 which means one Agent could hold multiple different role instances of the same role.

## 4 Case Analysis

### 4.1 Conference Management

Conference management system is a typical open system [1]. In the real world, ISAS/SCI conference had always to undergo serious and unexpected re-thinking of its organization [5]. In this paper, a model of FRAG methodology was compared with two situations: when the conference size is rather small and when the conference size becomes larger. AR was generated to represent the hierarchy of organizational rules.

In the process of analyzing and designing Conference management system, instead of creating all the models (e.g. environment models, interaction models, Agent model, service model and organizational rule model), this paper focuses on the FRS model, ORS model and ARS model of Conference management system, and especially illustrating how the FRAG solves the problem of adaptive system. This indicates that FRAG is well adapted of developing adaptive systems in open environments.

### 4.2 Organization Specification

The conference process is divided to four stages: Submission, Reivew, Decision, and Final paper collection follow the analysis of Scott [16]. Only three stages is concerned in this paper. FRS model is used in the phase of Organization Specification.

$$FRS = \langle \Sigma_{FR}, \Theta_{\sigma_{Ag}}(FR) \rangle$$

$$\Theta_{Ag}(\text{Organization}) = \langle \{\emptyset\}, \{\text{Submission, Review, Decision}\} \rangle$$

$$\Theta_{Ag}(\text{Submission}) = \langle \{\text{Organization}\}, \{\text{author, DB}\} \rangle$$

$$\Theta_{Ag}(\text{Author}) = \langle \{\text{Submission}\}, \{\emptyset\} \rangle$$

$$\Theta_{Ag}(\text{DB}) = \langle \{\text{Submission}\}, \{\emptyset\} \rangle$$

$$\Theta_{\sigma_{Ag}}(\text{Review}) = \langle \{\text{Organization}\}, \{\text{Partioner, Assigner, Reviewer}\} \rangle$$

$$\Theta_{Ag}(\text{Partioner}) = \langle \{\text{Review}\}, \{\emptyset\} \rangle$$

$$\Theta_{Ag}(\text{Assigner}) = \langle \{\text{Review}\}, \{\emptyset\} \rangle$$

$$\Theta_{Ag}(\text{Reviewer}) = \langle \{\text{Review}\}, \{\emptyset\} \rangle$$

$$\Theta_{Ag}(\text{Decision}) = \langle \{\text{Organization}\}, \{\text{Collector, Decision\_Make}\} \rangle$$

$$\Theta_{Ag}(\text{Collector}) = \langle \{\text{Decision}\}, \{\emptyset\} \rangle$$

$$\Theta_{Ag}(\text{Decision\_Make}) = \langle \{\text{Decision}\}, \{\emptyset\} \rangle$$

$\Sigma_{FR} = \{\text{Organization, Submission, Author, DB, Review, Partioner, Assigner, Reviewer, Decision, Collector, Decision\_Make}\}$

$$\Sigma_{AtR} = \{\text{Author, DB, Partioner, Assigner, Reviewer, collector, Decision Make}\}$$

### 4.3 Identifying Role and Relation

ORS and ARS are used in the phase of Identifying Role and Relation. F-R mapping is implemented through aggregating OR from FR.  $A \leftarrow \{B, C\}$  denoted A is aggregated from B and C,  $A \leftarrow \{B, C\}$  denoted B and C is inherited from the same ancestor A.



The change of size of Conference causes the change of the structure of Conference. Therefore two situations are considered in this paper:

1. When the conference organizers expect a limited number of submissions, the number of the roles used in this situation is relatively small and the functions that each role takes are relatively more;
2. When the number of submissions is much higher than expected, new roles are needed to be introduced in and the functions that role take are more refined.

#### 4.3.1 When the Conference Organizers Expect a Limited Number of Submissions

OR is generated as below:

$$PC\_Chair \leftarrow \{Partioner, Assigner, Decision\_Maker\}$$

$$\Theta_{Ag}(PC\_Chair) = \langle \{\emptyset\}, \{Partioner, Assigner, Decision\_Maker\} \rangle$$

$$PC\_Member \leftarrow \{Reviewer, Collector\}$$

$$\Theta_{Ag}(PC\_Member) = \langle \{\emptyset\}, \{Reviewer, Collector\} \rangle$$

$$\Sigma_{OR} = \{PC\_Chair, PC\_Member\}$$

AR is generated as below:

$$PC \leftarrow \{PC\_Chair, PC\_Member\}$$

$$\Theta_{in}(PC\_Chair) = \langle \{PC\}, \{\emptyset\} \rangle$$

$$\Theta_{in}(PC\_Member) = \langle \{PC\}, \{\emptyset\} \rangle$$

$$Organization\_Participint \leftarrow \{PC, DB\}$$

$$\Theta_{in}(PC) = \langle \{Organization\_Participint\}, \{PC\_Chair, PC\_Member\} \rangle$$

$$\Theta_{in}(DB) = \langle \{Organization\_Participint\}, \{\emptyset\} \rangle$$

$$All \leftarrow \{Organization\_Participint, Author\}$$

$$\Theta_{in}(Organization\_Participint) = \langle \{All\}, \{PC, DB\} \rangle$$

$$\Theta_{in}(Author) = \langle \{All\}, \{\emptyset\} \rangle$$

$$\Sigma_{AR} = \{PC, Organization\_Participint, All\}$$

#### 4.3.2 When the Number of Submissions Is Much Higher Than Expected

OR is generated as below:

$$PC\_Chair \leftarrow \{Partioner, Decision\_Maker\}$$

$$\Theta_{Ag}(PC\_Chair) = \langle \{\emptyset\}, \{Partioner, Decision\_Maker\} \rangle$$

$$PC\_Member \leftarrow \{Assigner, Collector\}$$

$$\Theta_{Ag}(PC\_Member) = \langle \{\emptyset\}, \{Assigner, Collector\} \rangle$$

$$Reviewer \leftarrow \{Reviewer\}$$

$$\Theta_{Ag}(Reviewer) = \langle \{\emptyset\}, \{Reviewer\} \rangle$$

$$\Sigma_{OR} = \{PC\_Chair, PC\_Member, Reviewer\}$$

AR is generated as below:

$$PC \leftarrow \{PC\_Chair, PC\_Member\}$$

$$\Theta_{in}(PC\_Chair) = \langle \{PC\}, \{\emptyset\} \rangle$$

$$\Theta_{in} (PC\_Member) = \langle \{PC\}, \{\emptyset\} \rangle$$

$$\text{Organization\_Participint} \leftarrow \{PC, \text{Reviewer}, DB\}$$

$$\Theta_{in} (PC) = \langle \{\text{Organization\_Participint}\}, \{PC\_Chair, PC\_Member\} \rangle$$

$$\Theta_{in} (\text{Reviewer}) = \langle \{\text{Organization\_Participint}\}, \{\emptyset\} \rangle$$

$$\Theta_{in} (DB) = \langle \{\text{Organization\_Participint}\}, \{\emptyset\} \rangle$$

$$\text{All} \leftarrow \{\text{Organization\_Participint}, \text{Author}\}$$

$$\Theta_{in} (\text{Organization\_Participint}) = \langle \{\text{All}\}, \{PC, DB\} \rangle$$

$$\Theta_{in} (\text{Author}) = \langle \{\text{All}\}, \{\emptyset\} \rangle$$

$$\Sigma_{AR} = \{PC, \text{Organization\_Participint}, \text{All}\}$$

#### 4.4 Assign Agent Role

When the conference organizers expect a limited number of submissions, Role PC\_Member has three Role Instance: I\_PC\_Member1, I\_PC\_Member2, I\_PC\_Member3.

R-A mapping as below:

Assign I\_PC\_Member1 to Agent1;

Assign I\_PC\_Member2 to Agent2;

Assign I\_PC\_Member3 to Agent3;

The organizational structure is one aspect of the system that is likely to be affected by the size of conference. When the number of submissions is much higher than expected, conference management will introduce a new OR named Review. FRAG can clearly distinguish function of Reviewer from organization and create a new OR named Review by F-R mapping, and then FRAG can explicitly support continuous design for adaptation/change system. AR PC is a representation of program committee while AR Organization\_Participant is a representation of organization participants. So the hierarchy of organizational rules can implemented by use of AR. Conference management can assign agent roles dynamically through R-A mapping, therefore implement open role space.

## 5 Conclusions

The use of FRAG, which is based on the improved role model of Gaia, is advocated as a methodology for developing adaptive systems in open environments. FRAG classifies roles into three categories: FR, OR, and AR. FR is the abstract correspondence to organizational function and FRS is used to describe the aggregative relation between functions; OR is participant of organization and ORS is used to describe the association relation between roles; AR is a representation of a role type and ARS is used to describe the inheritable relation between roles.

Furthermore, the ORRA process is proposed as a universal process of developing MAS by focus on the reason and intention that why different models are adopted by different methodologies. ORRA has three stages: organization specification, identifying role and relation, assigning role to agent, corresponding to three abstracts (function, role and agent) and two mappings (F-R mapping and R-A mapping) in

FRAG. F-R mapping explicitly implements the independence between organization function and organization structure. R-A mapping give a way of assigning agent role. The results of comparing the models of Gaia, MaSE, Roadmap, ARL and FRAG show that, although different models are adopted by different methodology, they can belong to a universal ORRA process.

The analysis of conference management shows that FRAG can accomplish the task of developing adaptive systems in open environments by clearly identify the role that going go to be added or deleted while the conference is undergoing unexpected re-thinking of its organization.

## References

1. Juan, T. and Sterling, L., The ROADMAP Meta-model for Intelligent Adaptive Multi-Agent Systems in Open Environments, Proceedings of the Fourth International Workshop on Agent Oriented Software Engineering, at AAMAS'03, Melbourne, Australia, July 2003.
2. Haiping Xu and Xiaoqin Zhang, "A Methodology for Role-Based Modeling of Open Multi-Agent Software Systems," Proceedings of the 7th International Conference on Enterprise Information Systems (ICEIS 2005), May 24-28, 2005, Miami, Florida, USA, pp. 246-253.
3. F. Zambonelli, N. R. Jennings and M. Wooldridge, "Developing multi-agent systems: the Gaia Methodology" ACM Trans on Software Engineering and Methodology 12(3), 2003.
4. Wooldridge, M., Jennings, N. R., and Kinny, D. 2000. The Gaia methodology for agent-oriented analysis and design. *J. Autonom. Agents Multi-Agent Syst.* 3, 3, 285–312.
5. Luca Cernuzzi, Franco Zambonelli, Dealing with Adaptive Multiagent Systems Organizations in the Gaia Methodology, Proceedings of the 2nd International Joint Conference on Autonomous Agents and Multi-Agent Systems (AAMAS'05), Utrecht, The Netherlands July 2005.
6. Zambonelli F., Jennings, N. R., AND WOOLDRIDGE, M. 2001 Organizational rules as an abstraction for the analysis and design of multi-agent systems. *Int. J. Softw. Knowl. Eng.* 11.3(Apr.). 303-328.
7. Carole Bernon, Massimo Cossentino, Marie Pierre Gleizes, Paola Turci, Franco Zambonelli: A Study of Some Multi-agent Meta-models. *AOSE 2004:* 62-77.
8. XinJun Mao, Ji Wang, Jijia Chen: Modeling Organization Structure of Multi-Agent System. Proceedings of the 2005 IEEE/WIC/ACM International Conference on Intelligent Agent Technology, Compiègne, France, September 2005. *IAT 2005:* 116-119.
9. Alexander Hämmerle, Anthony Karageorgos, Michael Pirker, Alois Reitbauer, Georg Weichhart 2004: A role-based infrastructure for customised agent system development in supply networks. *SMC (5) 2004:* 4692-4699.
10. Chernuchin, D., Lazar, O., Dittrich, and G.: Comparison of Object-Oriented Approaches for Roles in Programming Languages. In: 2005 AAAI Fall Symposium: Roles, an interdisciplinary perspective. (2005).
11. Wood M. F. and DeLoach S. A. An Overview of the Multiagent Systems Engineering Methodology. The First International Workshop on Agent-Oriented Software Engineering (AOSE-2000), 2000.
12. Bent Bruun Kristensen: Object-Oriented Modeling with Roles. Proceedings of 1995 International Conference on Object Oriented Information Systems, December 1995, Dublin, Ireland *OOIS 1995:* 57-71.

13. Ralph Depke, Reiko Heckel, Jochen Malte Küster: Roles in Agent-Oriented Modeling. *International Journal of Software Engineering and Knowledge Engineering* 11(3): 281-302 (2001)
14. Juan, T., Pearce, A. and Sterling, L., ROADMAP: Extending the Gaia methodology for Complex Open Systems, *Proceedings of the First International Joint Conference on Autonomous Agents and Multi-Agent Systems (AAMAS 2002)*, Bologna, Italy, July 2002.
15. Salaheddin J. Juneidi, George A. Vouros: Agent role locking (ARL): theory for multi agent system with e-learning case study. *IADIS AC 2005*: 442-450.
16. Scott A. DeLoach, "Modeling Organizational Rules in the Multiagent Systems Engineering Methodology" *Proceedings of the 15th Canadian Conference on Artificial Intelligence (AI'2002)*. Calgary, Alberta, Canada. May 27-29, 2002.

# Multi-modal Services for Web Information Collection Based on Multi-agent Techniques

Qing He<sup>1</sup>, Xiurong Zhao<sup>1,2</sup>, and Sulan Zhang<sup>1,2</sup>

<sup>1</sup>The Key Laboratory of Intelligent Information Processing,  
Department of Intelligence Software, Institute of Computing Technology,  
Chinese Academy of Sciences,  
PO Box 2704-28, Beijing, 100080, China

<sup>2</sup>Graduate University of Chinese Academy of Sciences  
{heq, zhaoxr, zhangsl}@ics.ict.ac.cn

**Abstract.** With the rapid information growth on the Internet, web information collection is becoming increasingly important in many web applications, especially in search engines. The performance of web information collectors has a great influence on the quality of search engines, so when it comes to web spiders, we usually focus on their speed and accuracy. In this paper, we point out that customizability is also an important feature of a well-designed spider, which means spiders should be able to provide multi-modal services to satisfy different users with different requirements and preferences. And we have developed a parallel web spider system based on multi-agent techniques. It runs with high speed and high accuracy, and what's the most important, it can provide its services in multiple perspectives and has good extensibility and personalized customizability.

## 1 Introduction

The rapid increase in information on the web has led to more and more difficulties in retrieving the exact information we really need, and there is no substitute for search engines when we are surfing on the Internet. As the most important component of search engines, web information collectors, also known as spiders, crawler, robots, worms and wanders, have become the primary means to obtain information, which are also widely used in other web applications such as web data mining, business information services and so on.

Due to such a big demand, many works have been done on web information collection and a lot of progress has been made [1,2,3,4,5]. To achieve high speed and efficiency in vast amount of webpages, it is a natural choice to parallel the spider's crawling process, that is to say multi-agent techniques become popular. For example, a parallel web spider model has been presented in [6], which is knowledge-based, distributed, configurable, modular and integrated. And in [7], the dynamic assignment mechanism is used to wipe off redundant webpages caused by parallelization, which is effective to improve the quality of fetched webpages.

For a web information collection system, we pursue its high speed, high quality, high efficiency etc. Besides all these features, to improve its usability, a good spider should be able to satisfy different users according to their own requirements and

preferences, which means that it should provide its services in multiple perspectives. In this paper, based on multi-agent theory and technology, we have developed a parallel web information collection system that runs fast and efficiently, and what's the most important, it provides multi-modal services in order to meet the needs of different individuals. For example, users can run it either in the background as a software toolkit or in the foreground as a kind of web service, and which searching mechanism or running mode is employed is also decided by users' actual demands and so on, which will be detailed in the following parts.

This paper is organized as follows. In Section 2, we give an overview of our spider model, including its architecture and some techniques employed. In Section 3, the multiple services provided by our web spider system are presented in detail. In Section 4, we brief the implementation and carry out some experiments, followed by our conclusions in Section 5.

## 2 An Overview of Our Spider Model

The parallel model for our web information collection system is presented in paper [6] as shown in Figure 1. As we can see, there are four kinds of agent in this model: facilitator, URL agent, spider agent and index agent.

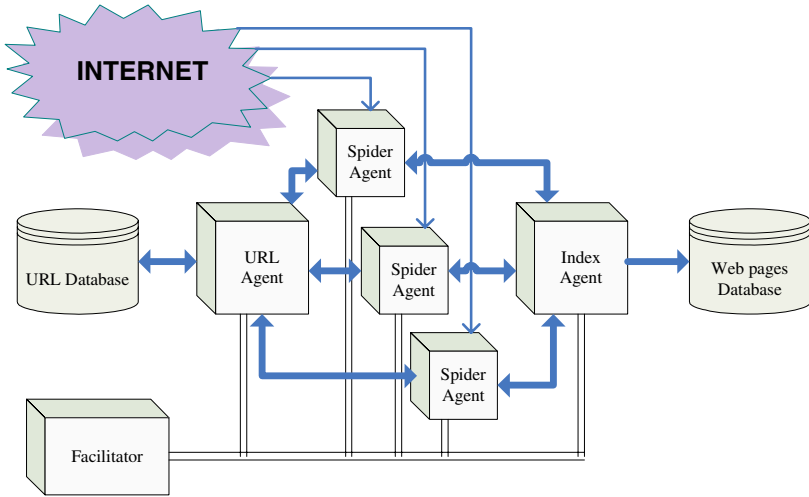


Fig. 1. Architecture of parallel web spider

Facilitator is a special agent in charge of resources and other agents in the system. It acts like the head of the body dominating the other organs, which can start, schedule or kill the other agents, and coordinate multi-agent actions. URL agent is the management center for URL tasks. It takes charge of fetching, sorting, distributing and coordinating URLs. On one hand, URL agent fetches URLs from URL database into the task queue and distributes them to spider agents. On the other hand, it

receives new URLs from spider agents, then coordinates and saves them into URL database. Spider agent is the main executive unit that receives URLs from URL agents, fetches webpages and parses their hyperlinks, and then sends these new URLs to URL agent and requests for new tasks. Many spider agents run in parallel under the control of the facilitator. The main job of index agent is indexing webpages and saving them into webpages database.

### 3 Several Important Sorts of Service

The basic function of our web information collection system is to filter and download webpages according to the parameters users set. The pre-set parameters mainly include keywords, number of spiders, start and end time of spiders, crawling mode, site type, file types of webpages, directory to save webpages and so on. Based on that, the system can provide multiple kinds of service, which makes it more flexible and adaptive to different situations. We'll introduce some of them in this part.

#### 3.1 Two Searching Mechanisms

Two searching mechanisms are employed in this system. One of them is the independent search based on initial URLs and the other is meta-search.

If we choose to use the independent searching mechanism, we should provide one or more initial URLs besides all the regular parameters beforehand. Then the spiders start from the initial URLs, filter webpages according to keywords and file types, parse hyperlinks on them, send new URLs to URL agent and request for new URLs. This mechanism can achieve better results if we provide proper initial URLs, so it works well under the instruction of prior knowledge.

Meta-search is another important kind of searching mechanism used in the process of web information collection. The basic idea of meta-search is making an interface between users' queries and other existing search engines, transforming users' queries and sending them to several search engines, then receiving searching results, organizing them in some way, and returning the final answers to users. In the study of meta-search method, we consult existing search engines with users' queries and obtain their searching results in the form of some webpages with URLs on them. As we all know, the searching results of good search engines are all ordered by webpages' importance or correlativity. The more important a webpage is, the closer to the head it is. So the webpages in the front are of more importance and correlativity, which we can choose as our starting points of the crawling process. Here we need to mention two important parameters that can influence the amount of webpages downloaded: width and depth. The parameter of width determines the number of URLs from the first searching result that will be used as the starting points and depth determines the number of levels spiders will trace down from a certain page. As we can see, different widths and depths will lead to different amounts of webpages. The panel of setting the two parameters is shown in Figure 2.

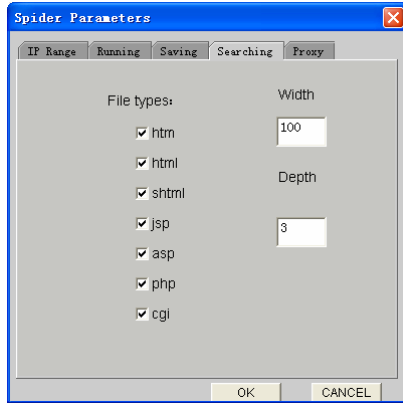


Fig. 2. The fourth parameter set panel of the system

### 3.2 Four Crawling Modes

In order to satisfy different requirements of searching and downloading, we have developed four crawling modes for spiders so that they can work in different ways and crawl proper pages according to our actual purposes. Figure 3 shows the four choices of crawling modes.

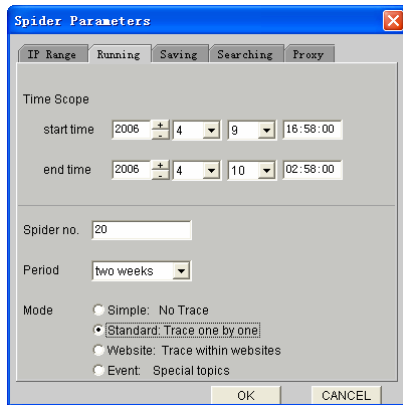


Fig. 3. The second parameter set panel of the system

Simple mode: In this mode, spiders only crawls webpages for given URLs without parsing hyperlinks on them. This mode works with high efficiency but with limited scope of applicability.

Standard mode: Spiders parse each hyperlink on each webpage and send the new URLs to URL agent. In another word, spiders will trace the hyperlinks one by one based on keywords with breadth-first strategy. This mode works slower than the simple mode



because it takes some time for spiders to parse hyperlinks, but it can download a large amount of webpages that we really want, thus this mode is widely used.

**Website mode:** Sometimes we need to search within some specific websites. In that case, we can choose website mode. In this mode, when a spider receives an URL of a website, the whole website will be considered as a unit and spiders will trace within one website after another. Technically, for a given website, each spider has two vectors to manage URLs. One stores inner URLs and the other stores outer URLs. After fetching a page, the spider will parse inner hyperlinks and outer hyperlinks and save them into their corresponding vectors. Then it will get new a URL from the inner vector to crawl another page and won't stop until all URLs in the inner vector are fetched. Namely it will stop after all the proper pages in this website are fetched. Since there may be many repeated hyperlinks in these pages in this website, the inner vector will filter them quickly. After a website is visited completely, the spider will get a new URL from the outer vector and repeat the cycle above.

**Event mode:** Here by “event” we mean some special topics on specific person or affair or something else that we're interested in. As we all know, there are more and more websites providing organized information about important events so that we can get to know them easily. If we want to download these webpages about these topics to do some further analysis and study, we select the event mode and provide the homepage as the initial URL. Then spiders will parse hyperlinks on it and fetch the corresponding pages about the topic. To illustrate this functionality, we have started spiders in this mode and fetched some pages about the Shenzhou VI Spaceship from internet website, which are all closely relative to the topic, then we process them and present the organized information about the topic as shown in Figure 4.

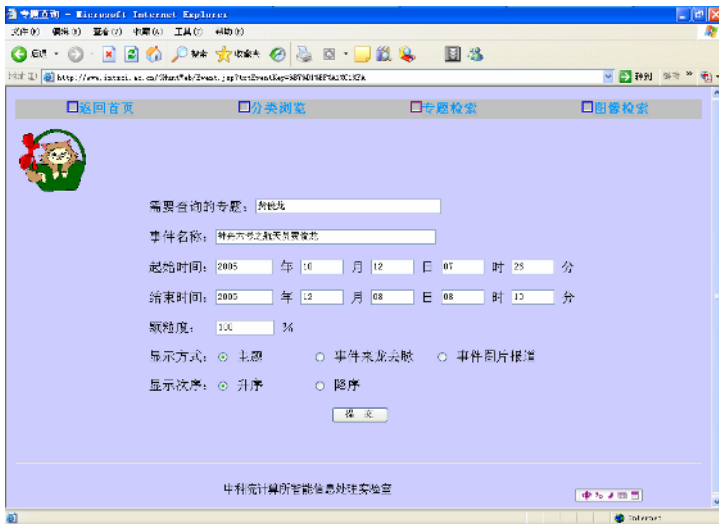


Fig. 4. Example of organized information after fetching pages in Event mode

### 3.3 Two Running Modes

To make it easier for users to access our tool of spider, we have published a web version as the foreground version in addition to the background version, so we can use it either as a desktop software toolkit or a kind of web service. The parameter set page of the web version is shown in Figure 5.

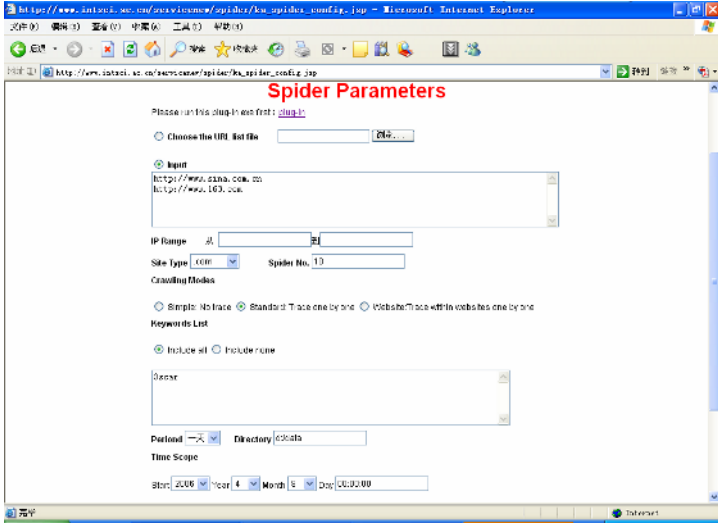


Fig. 5. The parameter set page of the web version

As for the foreground version, we locate the web spider system on a server and employ the typical C/S model, in which the clients send their requests to the server, and then the server starts spiders for crawling tasks and returns the crawled pages to the clients, with an embedded Java applet on client machines showing the statistical information of crawling processes in real time.

### 3.4 Other Important Services

There are several other important services that make it very convenient for users to use the system, like automatically updating and breakpoints resuming and so on.

Web information is changing every minute, with some pages being added or updated while some pages being removed. To make sure we always have the latest information, we have developed the function of automatically updating. Users can choose a proper period, for example two weeks, and then the spiders will be restarted automatically every two weeks so that our information is kept up to date.

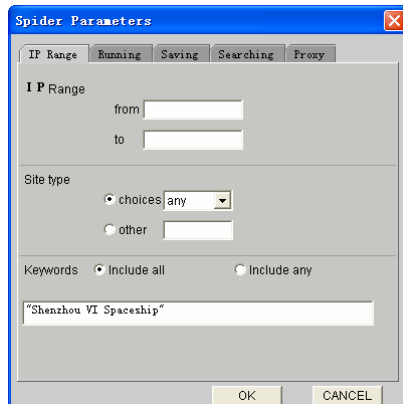
The performance of spider depends on the conditions of network a lot. But as we all know, the network condition is not always good, and sometimes it breaks off for some reason. In that case, restarting the spiders and re-crawling pages again from zero would be a great waste of time and resources. So the function of checking network conditions and resuming from breakpoints is very important, especially for data

transfer software working with support of networks. To improve speed of the crawling process, in our system, the task of inspecting network conditions is not done by spider agents themselves but by the facilitator. Every time before the facilitator starts up a spider agent, it will check up the network conditions first. If the network goes well, it will start up the spider agent as usual, else it will suspend all the running spiders until the network recovers. So when the network breaks off, spiders will sleep for a while and wake up to continue working till network recovery.

As described above, our spider system provides its functions in multiple perspectives and achieves personalized customizability to some extent so that different users can use it in different ways to meet their actual needs. Furthermore, the system is designed in a modular way and it is very easy to extend it with new functions. So the system is also with good extensibility.

## 4 Implementation and Experiments

The system is implemented in Java to be independent of operating systems. As described in Part 2, there are four kinds of agent in the system: facilitator, URL agent, spider agent and index agent. Each agent is implemented as a java class and the spider agent may have many threads running in parallel.



**Fig. 6.** The first parameter set panel of the system

Before starting spiders for crawling processes, we should set some parameters beforehand, which also reflect flexibility and customizability of the system. They mainly include IP range, site types, keywords, starting and ending time, number of spiders, running period, crawling mode, directory, file types, depth and width, proxy and so on. These parameters are shown in Figure 2, Figure 3 and Figure 6.

During the crawling process, spiders filter the webpages according to keywords. Spiders only fetch webpages that include all or any of the keywords. Therefore, the system runs with high accuracy. Now we give a test on the speed of spiders. As we all know, the speed of the crawling process is related to many factors, like the network

conditions, the number of spiders, the keywords, the crawling mode and so on. In the following experiment, the running environment of the system is Windows XP, with the CPU of Pentium IV 3GHZ, 512M memories and the network bandwidth is 100Mbps. We start multiple spiders to crawl pages about the information of Shenzhou VI Spaceship in the standard crawling mode. The results are shown in Table 1.

**Table 1.** The speed tests of our system

No. of Spiders	Time(s)	Pages	Size(MB)	Speed	
				Pages/m	Kb/s
20	3155	2122	92.6	40.8	31.2
100	1452	2209	65.1	92.1	47.0
500	720	1012	46.2	84.3	65.7

## 5 Conclusions

Web information collection is widely used in many web applications and we usually use criterions like speed and accuracy to evaluate web information collecting systems. We point out that customizability is also an important feature of a well-designed spider, which means spiders should be able to provide multi-modal services to satisfy different users with different requirements and preferences. Based on multi-agent theories and technologies, we have developed such a web spider system. Experimental results show that it runs with high speed and high accuracy, and what's the most important, it can provide its services in multiple perspectives, for example, users can choose searching mechanism, crawling mode and running mode according to their actual requirements. In conclusion, the system is both effective and convenient for users with different requirements and preferences.

## Acknowledgements

This work is supported by the National Science Foundation of China No. 90604017, 60435010, National Basic Research Priorities Programme No. 2003CB317004 and the Nature Science Foundation of Beijing No. 4052025.

## References

1. J. Cho, H. Garcia-Molina, and L. Page: Efficient Crawling Through URL Ordering. *Computer Networks and ISDN Systems*, 30 (1998) 161-172
2. S. Brin, L. Page: The anatomy of a large-scale hypertext Web search engine. *Computer Networks and ISDN Systems*, 30 (1998) 107-117
3. R.C. Miller, K. Bharat: SPHINX: A Framework for creating personal, site-specific Web crawlers. *Computer Networks and ISDN Systems*, 30 (1998) 119-130
4. M. Diligenti, F.M. Coetzee, S. Lawrence et. al.: Focused Crawling Using Context Graphs. In: *Proceedings of the 26th VLDB Conference*, Cairo, Egypt, 2000

5. M. Najork, J. L. Wiener: Breadth-first search crawling yields high-quality pages. In: Proceeding of 10th International World Wide Web Conference, 2001
6. Mingkai Dong, Shaohui Liu, Haijun Zhang, Zhongzhi Shi: Parallel Web Spider Based on Intelligent Agent. In: Proceedings of The 5th Pacific Rim International Workshop on Multi-Agents, Tokyo, 2002
7. Jiewen Luo, Zhongzhi Shi, Maoguang Wang, Wei Wang: Parallel Web Spiders for Cooperative Information Gathering. In: H. Zhuge, G.C. Fox (eds.): Grid and Cooperative Computing. Lecture Notes in Computer Science, Vol.3795. Springer-Verlag, Berlin Heidelberg Beijing (2005) 1192-1197
8. A. Heydon, M. Najork: Mercator: A Scalable, Extensible Web Crawler. World Wide Web, 2 (1999) 219-229
9. Honghui Peng, Zuoquan Lin: Search Engines and Meta Search Engines on Internet. Computer Science, 29 (2002) 1-12

# Formalizing Risk Strategies and Risk Strategy Equilibrium in Agent Interactions Modeled as Infinitely Repeated Games

Ka-man Lam and Ho-fung Leung

Department of Computer Science and Engineering  
The Chinese University of Hong Kong  
{kmlam, lhf}@cse.cuhk.edu.hk

**Abstract.** To design intelligent agents for multi-agent applications, like auctions and negotiations, we need to first analyze how agents should interact in these applications. Game theory is a tool, which can be used. In game theory, decision-making often depends on probability and expected utility. However, decision makers usually violate the expected utility theory when there is risk in the choices. Instead, decision makers make decisions according to their attitudes towards risk. Also, reputations of other agents in making certain actions also affect decision-making. In this paper, we make use of risk attitude, reputation and utility for making decisions. We define the concepts of *risk strategies*, *risk strategy equilibrium*, and a formalized way to find the risk strategy equilibrium in infinitely repeated games. Simulations show that players get higher payoff by using risk strategies than using other game theoretic strategies.

## 1 Introduction

A key property defining intelligent agent is social ability, which is the capability of an agent to interact with other agents or humans. So, before designing an intelligent agent for any multi-agent system, we have to first understand how agents should behave and interact in that particular application. This can be done by modelling the application as a game. Many multi-agent interaction problems, like auctions, negotiations, and bidding, can be modeled as games. To analyze these games and to understand how decision-makers interact, we can use a collection of analytical tools known as *Game Theory*.<sup>1</sup>

In game theory, decision-making often depends on probability and expected utility. Expected utility is the summation of the utility values of outcomes multiplied by their respective probabilities. According to the *Expected Utility Theory* [16], it is rational for a decision maker to make a choice with the highest expected utility. For example, there are two choices: one pays an utility of 3000 with probability of 1, while the other pays an utility of 4000 with probability of 0.8 and

---

<sup>1</sup> Due to lack of space, readers are assumed to be familiar with the Game Theory terminologies, or readers may refer to [11] for details.

otherwise nothing. According to the theory, decision makers should choose the latter one as the expected utility is higher. However, experiments [4] show that most of the decision makers violate the Expected Utility Theory when there is risk in the choices.<sup>2</sup> This is because decision makers have attitudes toward risk. The concept of risk attitude is also applied in multi-agent systems, like auctions [7], continuous double auctions [2], and trust/honesty model [6]. However, risk attitude is rarely considered in game analysis. At the same time, reputations of other agents in making certain actions also affect decision-making [14]. For example, if a person always choose to tell lies, it is less likely that his opponent will choose to believe him. However, reputation is also rarely considered in game analysis. In this paper, we make use of risk attitude, reputation and utility for making decisions in infinitely repeated games.

According to the theory of *Conditions of Learning* in psychology [1], attitudes will be changed by favorable or unfavorable experiences. In particular, people in threat situations become more rigid in taking risk [15]. For example, a person may become very risk-averse after losing a great sum in the stock market. So, a player should choose a suitable risk attitude as experience accumulated. In this paper, we define *risk strategy*, which is a function of experience that chooses a suitable risk attitude at each round of the game. Furthermore, a game is in equilibrium when each player's risk strategy is the best response to each others' risk strategies. We call this the *risk strategy equilibrium*, which will be defined formally in this paper.

The paper is structured as follows. We introduce the background of this work and a running example in sections 2 and 3 respectively. Then, we give definitions of basic concepts, risk strategies and risk strategy equilibrium in sections 4 and 5. In section 6, we compare the payoffs that the players get by using risk strategies and other strategies. We conclude and highlight future work in the last section.

## 2 Background

An important concept in game theory is *Nash equilibrium* [10], in which each player's strategy is the best response to each other players' strategies. In strategic games, we have *pure strategy Nash equilibrium*, in which each player's action is the best response to each other players' actions. However, not every strategic game has a pure strategy Nash equilibrium.

In that case, players may use *mixed strategies* or *behavioral strategies*. A mixed strategy is a lottery over all pure strategies. When each player's mixed (behavioral) strategy is the best response to each other players' mixed (behavioral) strategies, the outcome is called a *mixed (behavioral) strategy Nash equilibrium*.<sup>3</sup> However, the payoffs that the players get by using mixed or behavioral strategies may not be good enough when the game is repeated [5].

<sup>2</sup> In one of the experiments conducted by Kahneman and Tversky [4], out of 95 respondents, 80% choose the first choice, while only 20% choose the second one.

<sup>3</sup> Mixed strategies and behavioral strategies are similar and it is proved that every mixed strategy has an outcome-equivalent behavioral strategy [11].

An alternative is *trigger strategies*, with which players are agreed to play an action profile, and any players deviating from the agreement will be punished. Although a repeated game is in Nash equilibrium when all players use trigger strategies [8], it is possible that players who punish the deviated player may receive lower payoffs than not punishing the deviated player.

In [5], we proposed to use risk attitude and reputation for players to make decisions and we showed that risk strategies is a new concept, which is different from existing game theory concepts we mentioned above. However, definitions are limited to  $2 \times 2$  games. In this paper, we are going to generalize the definitions to infinitely repeated games, which involve multi-players and multi-actions.

### 3 A Running Example

Consider a strategic game with three players choosing their actions simultaneously. Players 1 and 2 choose whether to tell a lie or tell the truth, denoted as  $L$  and  $T$  respectively and player 3 chooses to believe the message from player 1, believe the message from player 2 or believe none of the messages, denoted as  $B_1$ ,  $B_2$  and  $N$  respectively. The game in extensive form is shown in Figure 1, where the payoffs  $a, b, c, d, e, f$  and  $g$  are 1, 0.5, 0.25,  $-0.25$ ,  $-0.5$ ,  $-0.75$ , and  $-1$  respectively. It can be verified that the game has no pure strategy Nash equilibrium and the mixed (or behavioral) strategy Nash equilibrium is a uniform probability distribution over the set of pure strategies, in which players 1 and 2 will both get an expected payoff of  $0.125t$ , while player 3 will get an expected payoff of  $-0.25t$  if the game is repeated for  $t$  rounds. Yet, players can get better payoffs by using risk strategies. This is illustrated in the following sections.

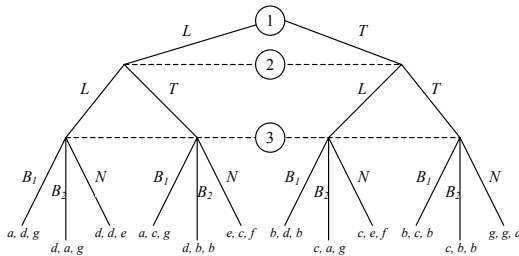


Fig. 1. The lying game in extensive form

### 4 Infinitely Repeated Games

A *strategic game* is a game in which the players choose their actions once and for all, and the players choose their actions simultaneously. When a strategic game is repeated infinitely, the game is called an *infinitely repeated game*.



**Definition 1.** Let  $G = \langle N, (A_i), (U_i) \rangle$  be a strategic game; let  $A = \times_{i \in N} A_i$ . An **infinitely repeated game** of  $G$  is an extensive game with perfect information and simultaneously moves  $\langle N, H, P, (\succsim_i) \rangle$ , in which

- $N = \{1, 2, \dots, n\}$  is the set of players
- $A_i$  is the set of available actions for player  $i$
- $U_i$  is the utility function for player  $i$ , where  $U_i(a_1, \dots, a_n)$  returns the payoff of player  $i$  in  $[-1, 1]$  for the action profile  $(a_1, \dots, a_n)$
- $H = \{\emptyset\} \cup (\cup_{t=1}^{\infty} A^t)$  is the set of histories, where  $\emptyset$  is the initial history
- $P(h) = N$  for each nonterminal history  $h \in H$  is the player function
- $\succsim_i$  is the preference relation for player  $i$

The  $k^{th}$  time that the constituent strategic game is repeated is called *round  $k$*  of the infinitely repeated game. In which, player  $i$  chooses action  $a_i^k$  and gets an utility of  $u_i^k$ , where  $u_i^k = U_i(a_1^k, \dots, a_n^k)$ . So, a sequence of utilities  $(u_i^1, \dots, u_i^k)$  forms an outcome  $O$  of the game. There are different ways to define preference relations  $\succsim_i$  over the outcomes [11]. In this paper, we use limit of means.

**Definition 2.** In an infinitely repeated game with **limit of means preference relation**, an outcome  $O_1$  with payoff sequence  $(p_i^k)$  is preferred by player  $i$  to another outcome  $O_2$  with payoff sequence  $(q_i^k)$ , denoted as  $O_1 \succsim_i O_2$ , if and only if  $\lim_{T \rightarrow \infty} \sum_{k=1}^T (p_i^k - q_i^k) / T > 0$ .

### 4.1 Risk Attitude and Reputation

In practice, humans do not make decisions only based on game analysis. Take gambling as an example, some players make risky decisions while some players do not. This is because players have their own experiences and their own attitudes towards risk. Some players are risk-averse while some are risk-seeking [4,14]. Also, people in general have different risk attitudes to different actions. One may be cheated by a hawker and bought rotten oranges, so he becomes risk-averse in buying oranges next time. However, this does not mean that he will be risk-averse in buying stocks in the stock market. To model this, we associate a *risk attitude* in choosing an action to each player at each round of the game. We define risk attitude for a player to choose an action be a real number in  $[0, 1]$ .

**Definition 3.** In an infinitely repeated game  $\langle N, H, P, (\succsim_i) \rangle$ , the **risk attitude of a player  $i$  in choosing action  $a \in A_i$  in round  $k$  of the game**,  $r_{i,a}^k$ , is a real number in  $[0, 1]$ . For  $r_{i,a}^k = 0, 0.5$ , and  $1$ , the player is said to be the most risk-averse, risk-neutral, and the most risk-seeking respectively.

To represent the experiences, we define *reputation* for each player in choosing an action be a real number in  $[0, 1]$ . Note that there are various ways to obtain reputations [9,12,13]. In this paper, reputation of a player in choosing action  $a$  (obtained in round  $k$ ) is given by the number of times that the player has chosen action  $a$  in the previous  $k - 1$  rounds of the game.

**Definition 4.** In an infinitely repeated game  $\langle N, H, P, (\succsim_i) \rangle$ , the **reputation of a player  $i$  in choosing action  $a \in A_i$  in round  $k$  of the game**,  $rep_{i,a}^k$ , is a real number in  $[0, 1]$ , where  $\sum_{a \in A_i} rep_{i,a}^k = 1$ .

### 4.2 Decision-Making

We suggest that in an infinitely repeated game, players can use their risk attitudes, reputations of other players, and the payoffs that will be brought by the actions to make decisions at each round of the game. Formally, player  $i$  chooses an action  $a_i \in A_i$  at round  $k$  which maximizes the returned value of a decision-making function  $f_d$ :

$$a_i = \arg \max_{a \in A_i} \mathbf{AVE}_{a_{-i}^* \in A_{-i}^*} (f_d(r, rep, u)), \text{ where}$$

- $A_{-i}^* = \{a_{-i}^* : a_{-i}^* = \arg \max_{a_{-i} \in A_{-i}} U_i(a_{-i}, a)\}$  is the set of action profiles of all players except  $i$  such that player  $i$  gets the maximum payoff with this action profile by choosing action  $a$
- $r = r_{i,a}^k$  is the risk attitude of player  $i$  in choosing action  $a$  in round  $k$
- $rep = \prod_{a_j, (a_j)_{j \in N \setminus \{i\}} = a_{-i}^*} rep_{j,a_j}^k$  is the product of reputations of all players  $j, j \in N \setminus \{i\}$ , in choosing action  $a_j$  at round  $k$  such that  $a_j$  is an action in the action profile  $a_{-i}^*$
- $u = U_i(a_{-i}^*, a)$  is the utility of player  $i$  for the action profile  $(a_{-i}^*, a)$
- $\mathbf{AVE}_{x \in X} (Y) = \frac{\sum_{x \in X} Y}{|X|}$  is an average function.

The function  $f_d$  must satisfy the following axioms:

1.  $f_d$  is continuous.
2. (*Adventurousness of risk-seeking agents*) There exists a value  $r_0 \in [0, 1]$  such that  $f_d(r, rep', u) > f_d(r, rep, u')$  if and only if  $r > r_0, rep > rep'$  and  $u > u'$ .
3. (*Cautiousness of risk-averse agents*) There exists a value  $r'_0 \in [0, 1]$  such that  $f_d(r, rep', u) < f_d(r, rep, u')$  if and only if  $r < r'_0, rep > rep'$  and  $u > u'$ .
4. If  $u \geq u', f_d(r, rep, u) \geq f_d(r, rep, u')$ .
5. If  $rep \geq rep', f_d(r, rep, u) \geq f_d(r, rep', u)$ .

It is obvious that the domains of the inputs of  $f_d$  are continuous, so  $f_d$  should be continuous. A widely adopted definition for risk-averse and risk-seeking in economics and psychology [3,4,14] is that a risk-averse person prefers getting something for sure, while a risk-seeking person prefers getting a higher utility even it is not for sure. In infinitely repeated games, players cannot get something for sure but the payoff that a player gets depends on the actions that other players choose. So, a risk-averse player chooses an action based on other players' reputations, assuming that a player has high reputation in choosing a particular action has a high probability in choosing the same action in the next round; while a risk-seeking player chooses an action with the highest payoff. This brings about axioms 2 and 3. Axiom 4 (axiom 5) states that if there are two actions with the same reputation (payoff), but with different payoffs (reputations), then the one with higher payoff (reputation) is preferred.

A simple example of decision-making function for player  $i$  in the lying game, satisfying the above axioms, can be defined as follows:

$$f_d = (1 - r_{i,a}^k) \times rep_{j,a_j}^k \times rep_{h,a_h}^k + r_{i,a}^k \times U_i(a_j^*, a_h^*, a) \tag{1}$$

where  $a_j^*, a_h^* = \arg \max_{a_j \in A_j, a_h \in A_h} U_i(a_j, a_h, a)$ . We will use this function through out the rest of the discussion.<sup>4</sup> With this function, it is possible that two or more actions result in the same decision value. This means that after considering the reputations and payoffs, the player thinks that these actions have the same attractiveness. In this case, the player can randomly choose among these actions. Players can also assign different probabilities to these actions, which will be discussed in future work due to lack of space.

Suppose at a certain instant of the infinitely repeated lying game,  $rep_{1,T}^k = rep_{2,L}^k = rep_{3,B_1}^k = rep_{3,B_2}^k = rep_{3,N}^k = 0.2$ ,  $rep_{1,L}^k = rep_{2,T}^k = 0.8$ , the decision-making of player 3 is shown in Table 1. Player 3 can get the highest payoff by choosing  $B_1$  if players 1 and 2 choose  $T$  and  $L$  respectively. So player 3 calculates the decision value for  $B_1$  by considering the reputation of player 1 in choosing  $T$ , the reputation of player 2 in choosing  $L$ , and the payoff that he can get in this case if he chooses  $B_1$ . Similarly, player 3 calculates the decision value for  $B_2$ . On the other hand, player 3 can get the highest payoff by choosing  $N$  if both players 1 and 2 choose  $L$  or if both of them choose  $T$ . For each of these two cases, player 3 calculates the decision value for  $N$ . Since the two cases will happen with the same probability, we take an average on the two decision values.<sup>5</sup> As the action  $B_2$  has the highest decision value, player 3 chooses  $B_2$  in this example.

**Table 1.** An example of decision-making in the infinitely repeated lying game

$a_3$	Decision values	
$B_1$	$(1 - r_{3,B_1}^k) \times rep_{1,T}^k \times rep_{2,L}^k + r_{3,B_1}^k \times U_3(T, L, B_1)$	$= 0.132$
$B_2$	$(1 - r_{3,B_2}^k) \times rep_{1,L}^k \times rep_{2,T}^k + r_{3,B_2}^k \times U_3(L, T, B_2)$	$= 0.612$
$N$	$\frac{1}{2} \times ((1 - r_{3,N}^k) \times rep_{1,L}^k \times rep_{2,L}^k + r_{3,N}^k \times U_3(L, L, N))$ $+ (1 - r_{3,N}^k) \times rep_{1,T}^k \times rep_{2,T}^k + r_{3,N}^k \times U_3(T, T, N)$	$= 0.178$

## 5 Risk Strategies and Risk Strategy Equilibrium

### 5.1 Risk Strategies

In human practice, risk attitudes can be changed by experiences [14]. Take gambling as an example again, some players stop gambling after losing a few times while some players do not stop even they have lost many times. This is because players have different ways to change their risk attitudes. In this paper, we define a *risk strategy* of a player to be a function assigning a risk attitude for each action at each round of the game.

**Definition 5.** A *risk strategy*  $\rho_i$ , of a player  $i$  in an infinitely repeated game  $\langle N, H, P, (\succsim_i) \rangle$  is a function that assigns a risk attitude  $r_{i,a}^k$  for each action  $a \in A_i$  at each round  $k$ .

<sup>4</sup> Note that any other functions satisfying the above axioms can be used.

<sup>5</sup> Note that it is possible that player 3 has extra information that the two cases mention above happen with different probability. If so, we take an expected value on the two decision values. Due to lack of space, this will be studied in future work.

An example of risk strategy for players 1 and 2 can be risk-averse in choosing  $L$  and risk-seeking in choosing  $T$  in the first round. While in the succeeding rounds, they become risk-averse in choosing an action if they chose that action and got negative payoff in the previous round. Otherwise, the risk attitudes remain unchanged. This can be expressed by equation 2, where  $i = 1, 2$ .

Another example of risk strategy for player 3 can be risk-seeking in choosing  $B_1$ , and risk-neutral in choosing  $B_2$  as well as  $N$  in the first round. While in the succeeding rounds, he becomes risk-averse in choosing the action if he chose the action and got negative payoff in the previous round and he exchanges the risk attitudes for  $B_1$  and  $B_2$  in each round. Otherwise, the risk attitudes remain unchanged. This can be expressed by equation 3.

$$r_{i,a}^k = \begin{cases} 0 & k = 1, a = L \\ 1 & k = 1, a = T \\ 1 & k > 1, a = L, a_i^{k-1} = T, \\ & u_i^{k-1} < 0 \\ 0 & k > 1, a = a_i^{k-1} = T, u_i^{k-1} < 0 \\ 0 & k > 1, a = a_i^{k-1} = L, u_i^{k-1} < 0 \\ r_{i,a}^{k-1} & \text{otherwise} \end{cases} \tag{2}$$

$$r_{3,a}^k = \begin{cases} 1 & k = 1, a = B_1 \\ 0.5 & k = 1, a = B_2 \\ 0.5 & k = 1, a = N \\ 0 & k > 1, a = a_3^{k-1}, u_3^{k-1} < 0 \\ r_{3,a}^{k-1} & k > 1, a = B_1, a_3^{k-1} = B_2 \\ r_{3,a}^{k-1} & k > 1, a = B_2, a_3^{k-1} = B_1 \\ r_{3,a}^{k-1} & \text{otherwise} \end{cases} \tag{3}$$

### 5.2 Risk Strategy Equilibrium

Suppose the players use the decision-making function as shown by equation 1, and the risk strategies as shown by equations 2 and 3 in the infinitely repeated lying game.<sup>6</sup> In the first round, players 1 and 2 both choose  $T$ , while player 3 chooses  $B_1$ , and get a payoff of 0.5, 0.25, and 0.5 respectively. Since players 1 and 2 gain, they do not change their risk attitudes. In the second round, they both choose  $T$  again, while player 3 chooses  $B_2$ . This pattern  $(T, T, B_1), (T, T, B_2), (T, T, B_1), (T, T, B_2), \dots$ , goes on as the game continues.<sup>7</sup> In this case, the players get the following payoffs in  $t$  rounds of the game:

- player 1:  $0.375t$  if  $t$  is even,  $0.375(t - 1) + 0.5$  if  $t$  is odd
- player 2:  $0.375t$  if  $t$  is even,  $0.375(t - 1) + 0.25$  if  $t$  is odd
- player 3:  $0.5t$

If players 1 and 2 use the risk strategy as shown by equation 2, the only way for player 3 to increase his payoff is to use other risk strategies, such that he chooses  $N$  instead of  $B_1$  or  $B_2$  in round  $k$  of the game. Then, players 1 and 2 will get a payoff of  $-1$  and player 3 will get a payoff of 1. As players 1 and 2 lose in round  $k$  by choosing  $T$ , they become risk-averse in choosing  $T$  and risk-seeking in choosing  $L$  in round  $k + 1$ . Then they both choose  $L$  in round  $k + 1$ . In this case, player 3 can at most get a payoff of  $-0.5$  by choosing  $N$ . As players 1 and 2 lose again in round  $k + 1$ , they keep choosing  $L$  in round  $k + 2$ , and so the best response for player 3 is to choose  $N$  in round  $k + 2$ . This pattern

<sup>6</sup> Note that at the first two rounds of the game, players lack information to calculate reputations of other players in choosing their actions. So, we take  $\frac{1}{x}$  be the reputations, where  $x$  is the number of available actions for the players.

<sup>7</sup> The triple denotes (player 1's action, player 2's action, player 3's action).

$(L, L, N), (L, L, N), \dots$  goes on as the game continues. As a result, although player 3 increases his payoff in round  $k$ , he loses in the subsequent rounds. So, he can only get a total payoff of  $0.5(k - 1) + 1 - 0.5(t - k)$  in  $t$  rounds, which is smaller than  $0.5t$ . So, player 3 cannot increase his payoff by using other risk strategies.

If players 2 and 3 use the risk strategies as shown by equations 2 and 3 respectively, the only way for player 1 to increase his payoff is to use other risk strategies, such that he chooses  $L$  instead of  $T$  in round  $k$  of the game, where  $k$  is odd. Then, player 3 will get a payoff of  $-1$  and player 1 will get a payoff of  $1$ . As player 3 loses in round  $k$  by choosing  $B_1$ , he becomes risk-averse in choosing  $B_1$  in round  $k + 1$ . Then player 3 chooses  $N$ , and player 2 still chooses  $T$  in round  $k + 1$ . In this case, player 1 can at most get a payoff of  $-0.5$  by choosing  $L$ . As player 3 loses again in round  $k + 1$ , he keeps choosing  $N$  in round  $k + 2$ , and so the best response for player 1 is to choose  $L$  in round  $k + 2$ . This pattern  $(L, T, N), (L, T, N), \dots$  goes on as the game continues. As a result, although player 1 increases his payoff in round  $k$ , he loses in the subsequent rounds. So, he can only get a total payoff of  $0.375(k - 1) + 1 - 0.5(t - k)$  in  $t$  rounds, which is smaller than  $0.375t$  and  $0.375(t - 1) + 0.5$ . So, player 1 cannot increase his payoff by using other risk strategies. Similarly, player 2 also cannot increase his payoff by using other risk strategies.

When all players are using their best risk strategies in response to the other players' best response risk strategies, we call this *risk strategy equilibrium*. In the above example, as all players cannot increase their payoffs by using other risk strategies, the game is in risk strategy equilibrium.

**Definition 6.** A risk strategy equilibrium of an infinitely repeated game  $\langle N, H, P, (\succsim_i) \rangle$  is a profile  $\rho^*$  of risk strategies with the property that for player  $i \in N$ , we have  $O(\rho_{-i}^*, \rho_i^*) \succsim_i O(\rho_{-i}^*, \rho_i)$  for every risk strategy  $\rho_i$  of player  $i$ .

### 5.3 Formulas for Finding Risk Strategy Equilibrium

When each player's risk strategy is the best response to every other player's risk strategy, the game is in risk strategy equilibrium. Formally, the best response risk attitude for each action at each round can be found by solving simultaneously the following set of equations for all  $i$ :

$$r_{i,a_i}^k = \arg \max_{r_{i,a_i}^k} (u_i^k + u_i^{k+1}), \text{ where} \tag{4}$$

- $u_i^k = U_i(a_1^k, \dots, a_n^k)$  is the payoff that player  $i$  gets in round  $k$
- $a_i^k = \arg \max_{a \in A_i} \mathbf{AVE}_{a_{-i}^* \in A_{-i}^*} (f_d(r_{i,a_i}^k, rep^k, U_i(a_{-i}^*, a)))$
- $rep^k = \prod_{a_j, (a_j)_{j \in N \setminus \{i\}} = a_{-i}^*} rep_{j,a_j}^k$
- $A_{-i}^* = \{a_{-i}^* : a_{-i}^* = \arg \max_{a_{-i} \in A_{-i}} U_i(a_{-i}, a)\}$

This set of equations find the best response risk attitudes for each player in each round by maximizing the payoffs that the player can get in two succeeding rounds. In the lying game, players 1 and 2 can maximize their payoffs in

two succeeding rounds by choosing  $T$ , and player 3 can maximize his payoff by choosing  $B_1$  and  $B_2$  alternatively. Although player 1 can increase his payoff by choosing  $L$  when player 3 chooses  $B_1$ , the reputation for player 1 to choose  $T$  will decrease and the reputation for player 1 to choose  $L$  will increase. By the definition of the decision-making function, when player 3 decides whether to choose  $B_1$ , he considers the reputation for player 1 to choose  $T$ . By axiom  $d_5$ , the decision-value for  $B_1$  decreases as reputation for player 1 to choose  $T$  decreases. Eventually, player 3 will not choose  $B_1$ . This is similar for player 2. As a result, player 3 will choose  $N$  eventually, then players 1 and 2 actually gets lower payoff by choosing  $L$  than choosing  $T$ . On the other hand, player 3 can increase his payoff by choosing  $N$  as players 1 and 2 choose  $T$ . However, the reputations for player 3 to choose  $B_1$  and  $B_2$  will then decrease and the reputation for player 3 to choose  $N$  will increase. Eventually, players 1 and 2 will not choose  $T$  and player 3 gets lower payoff as a result. So, in the first round, the best response risk attitudes for players 1 and 2 in choosing  $T$  is 1 and that in choosing  $L$  is 0, while the best response risk attitudes for player 3 in choosing  $B_1$  (or  $B_2$ ) is 1 and that in choosing  $B_2$  (or  $B_1$ ) and  $N$  is 0.5. Starting from the second round, the equations solve the best response risk attitudes with respect to the reputations in each round, while maximizing the payoffs that the player can get in two succeeding rounds.

**Theorem 1.** *Any risk strategy profile  $(\rho_1^*, \dots, \rho_n^*)$ , where  $\rho_i^* = (r_{i,a_i}^1, \dots, r_{i,a_i}^k)$ , such that  $(r_{i,a_i}^k)$  satisfies equation 4 for all  $i \in N$ ,  $a_i \in A_i$ , and  $k \in \mathbb{Z}^+$ , is the risk strategy equilibrium.*

By using equation 4, the risk attitude for one round is found by maximizing the utilities of two succeeding rounds. So for the whole game, a set of risk attitudes is found by maximizing the sum of the utility sequence  $(u_i^1, \dots, u_i^k)$ . As the sum of the utility sequence is maximized, the set of risk attitudes is the best response. So when equation 4 is solved simultaneously for all  $i$ , the sets of risk attitudes are in risk strategy equilibrium.

Theorem 1 states a sufficient condition for the risk strategy equilibrium. The question of whether there exists a risk strategy equilibrium in which the set of risk attitudes does not satisfy equation 4, will be studied in future work. Also, note that in game analysis, players do not need to use the same decision-making function and players do not need to know other players' decision-making functions. This is because although different decision-making functions may return different decision-values for the same action, the rankings of the decision-values for all the actions are actually the same since the decision-making functions satisfy the five axioms states in section 4. So, in finding the best response risk strategies, players do not need to know other players' decision-making functions but can use their own decision-making functions instead.

## 6 Simulation

Simulation is done to compare the payoffs that players get by using risk strategies and other game theoretic strategies, like pure strategies, mixed strategies, and

Player 3's strategies	Players 1 and 2's strategies						
	Risk	Trigger	Mixed	<i>AL</i>	<i>AL, AT</i>	<i>AT, AL</i>	<i>AT</i>
Risk	<u>375, 375, 500</u>	<u>375, 375, 500</u>	236, 246, -250	-247, -250, <b>-501</b>	-247, <u>499, 497</u>	<u>499</u> , -248, <b>498</b>	<u>375, 375, 500</u>
Trigger	<u>375, 375, 500</u>	<u>375, 375, 500</u>	-248, -248, -499	-249, -250, <b>-501</b>	-249, -249, -499	-249, -249, -499	<u>375, 375, 500</u>
Mixed	281, 275, -403	-249, -247, -488	126, 121, -251	166, 205, -888	110, 261, -277	233, 250, -444	54, 82, 311
<i>AB<sub>1</sub></i>	<u>999, 250, -997</u>	-248, -249, -499	749, 0, -249	<u>1000</u> , -250, -1000	<u>1000, 250</u> , -1000	500, -250, <b>500</b>	500, <u>250</u> , 500
<i>AB<sub>2</sub></i>	<u>250, 999, -997</u>	-249, -248, -499	0, 750, -250	-250, <u>1000</u> , -1000	-250, 500, <b>500</b>	<u>250, 1000</u> , -1000	<u>250</u> , 500, 500
<i>AN</i>	<u>-125, -125, -750</u>	-250, -250, -499	-375, -375, -249	-250, -250, <b>-500</b>	-500, <u>250</u> , -750	<u>250</u> , -500, -750	-1000, -1000, <b>1000</b>

**Fig. 2.** The average results of 1000 repeated lying games, each has 1000 rounds

trigger strategies. The lying game, as shown in Fig 1, is simulated. There are five pure strategies that players may use: players 1 and 2 may always choose *L* or always choose *T*, denote as *AL* and *AT* respectively; while player 3 may always chooses *B<sub>1</sub>*, always chooses *B<sub>2</sub>*, or always chooses *N*, denote as *AB<sub>1</sub>*, *AB<sub>2</sub>*, and *AN* respectively. For players using mixed strategies, they play the one in mixed strategy equilibrium: players 1 and 2 choose *L* and *T* with probability  $\frac{1}{2}$ , and player 3 chooses *B<sub>1</sub>*, *B<sub>2</sub>* and *N* with probability  $\frac{1}{3}$ . For players using trigger strategies, players 1 and 2 agree to play *T* and player 3 agrees to play *B<sub>1</sub>* and *B<sub>2</sub>* alternatively. If anyone deviate from the agreed action profile, players 1 and 2 will play *L* and player 3 will play *N* until the end of the game. For players using risk strategies, they use equation 1 for decision-making, and they use the risk strategies as shown by equations 2 and 3.

In game theoretic analysis, players choose their best response strategies given other players' strategies. However, other players' strategies are often unknown in practice. Following the real situations, players are not informed with other players' strategies in this simulation. In each game in the simulation, the lying game is repeated for 1000 rounds. The average results of 1000 games are shown in Figure 2.<sup>8</sup>

This simulation shows the limitations of traditional game theoretic strategies. First, the payoffs get by using mixed strategies are not good enough. In fact, using risk strategies can get higher payoffs. So, players should not make decisions only based on probabilities and expected utilities. Second, players can get high payoffs by using trigger strategies only if other players do not deviate from the agreed action profile. Otherwise, the payoffs are extremely low because players get negative payoffs when they punish the deviated players. Third, players can get high payoffs by using pure strategies only if other players choose the same actions in every round of the game. However, it is irrational for players to do so since there is no pure strategy Nash equilibrium in the lying game. Also, a player can choose a suitable pure strategy only if other players' actions are known beforehand, which is impossible in practice.

<sup>8</sup> Payoffs are rounded off to the nearest integer. The triple in each cell of the table denotes the payoffs of players 1, 2, and 3. The payoffs which are underlined are the highest payoffs that players 1 and 2 get for each strategy that player 3 uses, while those that are bolded are the highest payoffs that player 3 gets for each strategy that players 1 and 2 use.

In general, players get the highest payoff by using risk strategies, no matter what strategies other players use, except that when other players use mixed strategies. In fact, no strategy can outperform in this case because mixed strategies are probabilistic in nature and the actions are randomly chosen. On the other hand, when interacting with players using pure strategies, players using risk strategies can get payoffs which are comparable to the payoffs obtained by using the best response pure strategies. This is because reputation is considered in decision-making. When other players always choose particular actions, the reputations increase. This enables players using risk strategies choose suitable actions, even if other players' actions is unknown. We can see that risk strategies outperform traditional game theoretic strategies like pure strategies, mixed strategies, and trigger strategies.

## 7 Conclusions and Future Work

We can use game theory to analyze multi-agent interaction problems and design intelligent agents. However, the concept of risk attitude is not defined in game theory, which it is sometimes necessary for analyzing agent interactions. In many games, pure strategies Nash equilibrium does not exist. In game theory, decision-making depends on probability and expected utility, like mixed or behavioral strategies. However, they are not good enough when the game is repeated. Although a repeated game is in Nash equilibrium when players use trigger strategies, trigger strategies is not always good for the players as players may get a lower payoff by punishing the deviating players. To solve the problems, we introduce *risk strategy*. Players using risk strategies make decisions with their experience, risk attitudes as well as utilities. We find that players can get better payoffs by using risk strategies than using mixed or behavioral strategies in infinitely repeated games. In addition, we find that a game without pure strategy Nash equilibrium can converge to a new type of equilibrium, which we define as *risk strategy equilibrium*. Also, simulation shows that players using risk strategies outperform players using other game theoretic strategies.

This paper only introduces the concepts of risk strategies and risk strategy equilibrium. In fact, there are more to study. For example, properties of risk strategies and risk strategy equilibrium. Existence of risk strategy equilibrium for different types of games. Conditions for existence of risk strategy equilibrium. Also, we shall use the proposed concepts to study multi-agent problems like auction, bidding, and the prisoners' dilemma, etc.

## References

1. R. Gagne. *The Conditions of Learning*. New York: Holt, Rinehart and Winston, 1985.
2. M. He, H. F. Leung, and N. Jennings. A fuzzy-logic based bidding strategy for autonomous agents in continuous double auctions. *IEEE Transactions on Knowledge and Data Engineering*, 15(6):985–1003, 2003.



3. <http://www.amosweb.com/gls>. Amosweb economic gloss.
4. D. Kahneman and A. Tversky. Prospect theory: An analysis of decision under risk. *Econometrica*, 47(2):263–291, 1979.
5. K. M. Lam and H. F. Leung. Risk strategies and risk strategy equilibrium in agent interactions modeled as normal repeated  $2 \times 2$  risk games. In *The Eighth Pacific Rim International Workshop on Multi-Agents*, 2005. Paper received the Best Paper Award.
6. K. M. Lam and H. F. Leung. A trust/honesty model with adaptive strategy for multiagent semi-competitive environments. *Autonomous Agents and Multi-Agent Systems*, To appear.
7. Y. Liu, R. Goodwin, and S. Keonig. Risk-averse auction agents. In *Proceedings of Autonomous Agents and Multi-Agent Systems*, pages 353–360, 2003.
8. R. D. Luce and H. Raiffa. *Games and Decisions*. New York: John Wiley and Sons, 1957.
9. L. Mui, M. Mohtashemi, and A. Halberstadt. A computational model of trust and reputation. In *Proceedings of 35th Hawaii International Conference on System Science*, 2002.
10. J. F. Nash. Equilibrium points in n-person games. In *Proceedings of the National Academy of Science of the United States of America*, pages 48–49, 1950.
11. M. J. Osborne and A. Rubinstein. *A Course in Game Theory*. MIT Press, 1994.
12. J. C. Rubiera, J. M. M. Lopez, and J. D. Muro. A fuzzy model of reputation in multi-agent systems. In *Proceedings of the Fifth International Conference on Autonomous Agents*, pages 25–26, 2001.
13. J. Sabater and C. Sierra. Regret: A reputation model for gregarious societies. In *Proceedings of Fourth International Workshop on Deception, Fraud and Trust in Agent Societies*, 2001.
14. A. Q. Sartain, A. J. North, J. R. Strange, and H. M. Chapman. *Psychology — Understanding Human Behavior*. McGraw-Hill Book Company, 1962.
15. B. M. Staw, L. E. Sandelands, and J. E. Dutton. Threat-rigidity effects in organizational behavior: A multilevel analysis. *Administrative Science Quarterly*, 26:501–524, 1981.
16. J. von Neumann and O. Morgenstern. *Theory of Games and Economic Behavior*. Princeton University Press, 1944.

# Reverse Auction-Based Grid Resources Allocation

Zhengyou Liang<sup>1</sup>, Yu Sun<sup>1</sup>, Ling Zhang<sup>2</sup>, and Shoubin Dong<sup>2</sup>

<sup>1</sup> School of Computer and Electronic Information, GuangXi University,  
NanNing, 530004, P.R. China

Lzyfile@sohu.com, sunyu1225@163.com

<sup>2</sup> Guangdong Key Laboratory of Computer Network, South China  
University of Technology, GuangZhou, 510641, P.R. China  
{Ling, sbdong}@scut.edu.cn

**Abstract.** Resources allocation and tasks scheduling is key technology in grid computing system. The market-based resources allocation model is considered as a good one. In this paper, a resources allocation model, based on reverse auction, was proposed, and its mechanism and related pricing algorithms were designed. In this model, a resources consumer invites a public bidding on the basis of his deadline and budget, then a resources provider bids according to his load, and the bidder who bids the cheapest price will win the auction. Numerous simulating experiments based on our proposed model was conducted, the experiments showed that our model can satisfy a user's QoS demand on deadline and budget, and have better performance in user utility, society utility, load-balance, job-completed rate than a commodity market-based resources allocation model.

## 1 Introduction

Grid computing is a new hotspot in the field of computer research in recent years. Grid research in the past 10 years has focused on the fields of resources management, communication, security and data management, and has gained numerous achievements. In grid computing system, economic problem is an important element, which will bring and promote the grid computing to the mainstream computing technology [1]. The economic problems frequently met in the grid computing include: how to encourage a grid resources provider to contribute computing resources and how to maintain them, how to encourage a grid resources consumer to make rational use of resources, and how to better allocate grid resources. Although some researches have been conducted on this aspect, many more researches need to be done in depth and width. Those economic problems mentioned above will be an important aspect in the field of grid research in the next decade [2].

There are some researches on the economy-based resources allocation. Buyya et al proposed an economy-based framework, investigated and developed an economic-based middleware and application system [3,4,5,6,7,8,9,10]. In his framework, the market model was commodity market, whose price of resources is constant in long term. As the constant price can't dynamically reflect the change of supply-demand relationship in the grid computing system, the resources allocation isn't optimized.

In this paper, we took reverse auction mechanism into grid resources allocation. Reverse auction is a dynamic pricing method that can reflect the supply-demand

relationship and the resources' value in time. We proposed a reverse auction mechanism and designed a method of how to calculate the resources' price according to its workload. A simulating system was developed. A lot of simulating experiments have been conducted based on our proposed model. The simulating experiments show that the reverse auction-based resources allocation had better performance on user utility, workload balance, job-completed rate and societal utility than commodity market.

## 2 Market-Based Resources Allocation Model

In a grid computing system, resource consumers (Consumer, User) and providers (Resources Owner, Provider) have different objectives, strategies, and supply-and-demand patterns. And in resources management system, scheduling decisions are mainly driven by system-centric parameters or driven by the Users requirements. Patterns which driven by system-centric parameters, as traditional approaches to manage resources, aim to enhance the system utility. They are used in a single administration domain. Patterns which driven by the Users requirements aim to maximize the QoS-based utility. In the latter patterns, one or some performance attributes which Users regarded as important will be optimized. As it is needed to compensate the system when performing the QoS demand, the method driven by the User's requirements is economic-based [4]. The benefits of economics-based resource allocation had discussed in [1].

In economics-based grid, market plays a basic role in the allocation of grid resources. Process of grid resources allocation is shown in **Fig. 1** [11]. In this process, a User Agent represents a User. User Agent finds and selects part of or all resources providers for executing User's job according to User's requirement, negotiates prices with Resources Providers, submits User's tasks, and finally returns the result of executed job to Users. The process of resources allocation can be described as follows:

- 1) User submits to User Agent his job constrained by deadline and budget, data files and pricing strategy.
- 2) User Agent inquires the Grid Information Server (GSI) about the available Resources Providers according to User's pricing strategy.
- 3) The GSI returns a set of available Resources Providers to User Agent.
- 4) If no Resources Providers are available, User Agent informs User of non-available computing resources. The allocation of grid resources ends.
- 5) User Agent selects some Resources Providers according to his pricing strategy. And User Agent and Resources Providers come to a resources price according to the market mechanism. If a resources price is not reached, go back to step 4.
- 6) If a resources price is reached, the Resources Provider executes the User's job according to the trade protocol, and accounting cost of executing the job. With the job finished, the result will be returned to User Agent.
- 7) User Agent returns the result to User, and User pays for the service.

The security mechanism and communication mechanism that support the interaction between the User Agent and Resources Provider are supported by existing grid

middleware such as Globus Toolkit. To realize this process, the following problems need to be solved:

1) Market Mechanism Design. It is important to design a market model to make sure that User and Resources Provider can well and truly express their value and trade policy, enhance User’s utility, Provider income, social utility in the case of satisfying user’s Qos demand. Further more, this market model encourages the User to rationally use the grid resources and the Resources Owner to add his computing resources to the global grid to gain profit by providing service.

2) Design User Agent and Resources Provider. User Agent represents a User to bargain with the Resources Provider in a global grid, and submits his job to the Resource Providers who agree to a trade protocol. User Agent maximizes his utility by selecting suitable Resources Providers. Usually, the User’s job needs the cooperating computing of many Resources Providers. User Agent should have the capability to negotiate with Resources Provider, and ensure the job to be finished before the deadline by allocating tasks to Resources Providers.

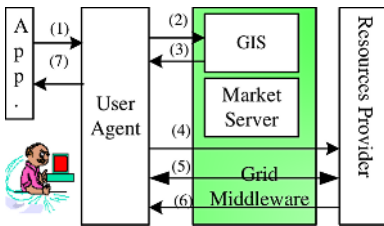


Fig. 1. Market-based Grid Resources Allocation

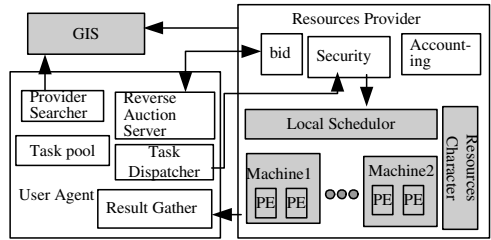


Fig. 2. Simulating System

Resources Provider needs to do the following: Firstly he performs job according to the trade protocol; the local task scheduler of Resources Provider should allocate enough resources to finish the tasks which user requires to be completed before the deadline. Secondly, the Resources Provider can dynamically change his service price according to the demand-supply relationship in the market and maximize his profit. Finally, in order to give a rational price and satisfy the User’s Qos demand, a Resources Provider should have the capability to estimate the resources usage and execution time for executing an application.

Accounting-Charging-Paying system, already discussed in [11], is needed to support the market trade. In this paper, only the market model and the resources allocation algorithm are discussed.

### 3 Reverse-Auction Mechanism Design

The auction mechanism, widely studied in economics and game theory, provides an effective and efficient solution to tasks and resources allocation in multi-agent system [12]. Reverse auction is an auction model driven by users requirements.

### 3.1 Reverse-Auction Protocol

An auction system usually consists of auctioneer, Resources Provider (bidder), Resources Consumer (tenderer), and arbitration institution. Besides these components, an E-economy auction system should have authentication institutions. In economy-based resources allocation system, scheduling and allocating decisions are driven by users requirements. In our system, the Consumer act as auctioneer. The Consumer notifies all available Resources Providers that he needs computing service, then each Resources Provider bids according to his price policy, and finally the Consumer decides who win the bid according to the auction rules. This form of auction is called reverse auction.

In our auction model, Consumers wishes to buy some computational service without knowing the service's value. Each bidder himself knows exactly what the value of goods is, and bids according to his own private price. Winner is the bidder with the lowest bid. The process is shown in **Algorithm 1**. Obviously, the mechanism presented in **Algorithm 1** is an incentive-compatible dominant-strategy mechanism.

**Algorithm 1.** First-price sealed-bid reverse auction

BEGIN

1. Consumer calculates reserve price based on his budget.
2. Consumer invites public bidding.
3. All providers submit their bids to the Consumer.
4. The winner is the Provider with the lowest bid that is lower than the Consumer's reserve price.
5. Consumer pays the winner the lowest bid.

END

In reverse-auction mechanism, a Provider's optimal strategy is to submit a bid equal to the goods' valuation. The strategy assumes that a player loses nothing by revealing her valuation to the principal and other players. The cost of revelation becomes important in scenarios with repeated interaction, because the principal can learn the players' valuations and get a substantial reverse price. The transaction will not occurs when all the bids are higher than the reserve price.

### 3.2 Rules for Grid Resources Consumers

As an auctioneer, Consumer expresses his willingness as follows:

$$\text{Buyer request } (i)=(J_i, \Gamma_i, JN_i, M_i) \quad (1)$$

Where  $i$  is Consumer ID,  $J_i$  is computational workload requested by Consumer  $i$ ,  $\Gamma_i$  is deadline of the job,  $JN_i$  is the number of tasks,  $M_i$  is machines attributes required by job, such as CPU speed, memory size, operating system. User has a secret reserve price  $RP_i$ . As job's budget is finite, one way to maximize User utility is to reduce computing service's cost. The private price of job is:

$$P_i^0 = \frac{B_i}{L_i} \quad (2)$$

Where  $B_i$  is the job's budget,  $L_i$  is the task's length.  $RP_i$  is ensured to lower than  $P_i^0$  in order to ensure the cost of job is no more than budget.

$JN_i$  is an important parameter in computational market. Usually, when the job is a single serial task or a MPI parallel job, only one Provider is needed. In this case,  $JN_i$  is considered as 1. If the job is the parameter sweep job or a batch job, which is composed of numerous independent tasks, many Providers are needed to work together to finish the job in time. In this case, we take  $JN_i$  as the number of independent tasks.

### 3.3 Rules for Grid Resources Provider

Provider indicates his bid as follows:

$$Seller\ offer\ (j)=(W_j, JN_j, P_j) \quad (3)$$

Where  $j$  is the  $j$ th provider,  $W_j$  is the workload (task length) that the Provider can finish before the deadline,  $JN_j$  is the number of tasks that the provider can finish before the deadline;  $P_j$  is the price of service.

In the case of serial job or a MPI parallel job,  $W_j$  of the buyer must be the same as that of the seller. So is  $JN_i$ . For batch job or parameter sweep job which can be allocated to different Providers and cooperative calculating, a Provider decides  $W_j$  and  $JN_i$  according to his capability, and the left tasks can be carried out by other Providers.

It is important for the Provider to decide the bid because it affects the auction's result and consequently his income. Provider is usually considered a rational one that pursues maximizing his profit, so that he bids as high as possible. On the other hand, the bidder takes the private value as the bid in our reverse auction. So the bidder will bid according to the relationship of supply and demand. The greater the demand is, the higher the bid is. The less the demand is, the lower the bid is. Suppose the cost of service is  $P_j^0$ , and the profit margin is  $R_j$ , the bid price can be calculated as follows:

$$P_j = P_j^0 (1 + R_j) (W_{PE} * (BASE_j + PE_{-}F_j * \frac{JNR_j}{PES_j}) + W_S * (BASE_j + S_{-}F_j * \frac{LOAD_j}{SPEED_j})) \quad (4)$$

Where,  $PES_j$  is the CPU number of Provider,  $SPEED_j$  is the total computing rate of the Provider's local computing system,  $JNR_j$  is the task number that are executed or waiting to be executed in the local computing system of the Provider,  $PE_{-}F_j$  and  $S_{-}F_j$  is the adjusted coefficient of CPU load and workload corresponsive,  $W_{PE}$  and  $W_S$  is weight factor of CPU load and workload corresponsive, and  $W_{PE} + W_S = 1$ .  $BASE_j$  is the base price of bid.

### 3.4 Winner's Rule

After Providers bid, an auctioneer determines who wins according to the following rules:

- For a single serial task or a MPI parallel job, since only one Provider is needed, there is only one winner. The winner is the Provider with the lowest bid which is lower than Consumer's reserve price. Consumer pays the winner the lowest bid. If there is no winner, Consumer will place another auction at a right time or can place the auction again right away by increasing the budget.
- For batch job or parameter sweep job, one or more Providers are needed according to the job's workload, so there can be many winners. We define that there is only one winner in one auction. If the winner himself cannot finish the job, Consumer will place another auction for the left tasks until the entire job is allocated.

## 4 Experiments

### 4.1 Simulation System Design and Realization

We developed a reverse auction-based grid resources allocation simulation system by using Gridsim [13]. The system includes three components: User Agent, Grid Information System (GIS), and Provider's local computation system, shown in **Fig.2**. The local computation system provides computation power. The GIS, as a market directory, helps User Agents discover providers. The User Agent, representing User, finds the available Providers in the worldwide, acts as an auctioneer, and submits the task to the winners.

**Resources Provider.** A Provider's local computation system consists of simulation processor, simulation machine, simulation local scheduling system and other simulation resources infrastructure, which is provided by Gridsim. **Accounting** component, **Security** component and **bid** component, developed by us, support reverse auction.

**User Agent.** User Agent includes **Provider Searcher**, **Reverse Auction Server**, **Task Dispatcher**, **Result Gather** module, and a **task pool**. **Provider Searcher** gets information about available Providers from GIS. **Reverse Auction Server** places the bid for tasks, and calculates reserve price by using **expression (2)**, where  $B_i$  is the left budget and  $L_i$  is the length of unfinished tasks. Auction protocol is described as **Algorithm 1**. **Task Dispatcher** submits tasks to Providers according to auction's result. The **Result Gather** module gathers the result of submitted tasks from Providers.

### 4.2 Utility

**User Utility.** User utility depends on some parameters. Because User takes the job's deadline as an important QoS requirement, we use finish time as an output expectation of User's budget. We based user utility on deadline. Supposing a job's budget is  $B_i$ , deadline is  $\Gamma_i$ , User utility is defined as follows:

$$U_i = \frac{2B_i}{1 + \exp(k_i(\tau_i - \Gamma_i)/\Gamma_i)} - E_i \quad (5)$$

Where  $\tau_i$  is finish time of the job;  $k_i$  is a factor expressing the user expectation;  $E_i$  is compensation of the computational resources, that is, the cost paid to the Provider. Utility function illustrates that a User wishes to reduce the transacted price so that the cost of executing his job is lowered. Finish time  $\tau_i$  stands for the QoS, if  $\tau_i > \Gamma_i$ , the QoS is lower and the utility is lower, if  $\tau_i = \Gamma_i$ , the QoS achieves User's goal, and if  $\tau_i < \Gamma_i$ , the QoS is beyond User's expectation and the utility is high. **Expression (5)** better reflects the User's requirement that means User wishes to reduce the compensation  $E_i$  and the finish time  $\tau_i$ .

**Society Utility.** Resources allocation is a societal work. Consumers and Providers, as participations, are all seeking the maximization of one's utility. However, goals maximizing personal utility are conflicting one another. How to weigh the effect of market mechanism that allocates the resources? We define society utility described the effect of market mechanism:

$$U = \sum_{j=1}^n w_j \log(1 + RE_j) \quad (6)$$

Where  $w_j$  is weight of the  $j$ th Provider's importance, and  $\sum_{j=1}^n w_j = 1$ .  $RE_j$  is the  $j$ th Provider's income. We consider that each Provider's weight is the same, and  $w_j = \frac{1}{n}$ ,  $j \in [1, n]$ . When  $\sum_{j=1}^n RE_j = \text{constant}$  and every Provider's weight is same, if a resource allocation model make every Provider's income same, the society utility will be the largest. It is easy to prove this affirms and we pass over the proving.

Features of society utility are: as defined in **equation (6)**, when the total income is constant, if every Provider gain balance income by balance allocation, the society utility will be optimized. Society utility expresses the rule of wealth allocation—the wealth should be allocated to all individuals as evenly as possible. For grid system, in the case of constant workload, the workload should be allocated to all computing resources, in this sense; the society utility also expresses the idea of load-balance.

### 4.3 Experiments

**Grid Resources Model.** Resources models of our grid, developed on the basis of the model introduced in paper [13], are shown in detail in Table 1. The price in Table 1 is the cost of each CPU in unit time in commodity market model. Price of the CPU of the same kind can be different, if they belong to different providers. Resources price is constant in commodity market model. But resources price can be changed according to the supply-demand relationship in reverse auction-based market, and is calculated by expressions (4); in our experiment, value of  $W_{PE}$  and  $W_S$  is 0.5,  $BASE_j$  is 0.7,  $PE\_F_j$  is 0.4, and  $S\_F_j$  is 0.01.



**Table 1.** Resource Model[13]

Provider ID	Machine	CPU number	CPU Speed	Price
R0	Copaq,AlphaServer,CPU,OSF1	4	515	8
R1	Sun,Ultra,Solaris	4	377	4
R2	Sun,Ultra,Solaris	4	377	3
R3	Sun,Ultra,Solaris	2	377	3
R4	Intel,Pentium/VC820,Linux	2	380	2
R5	SGI,Origin 3200,IRIX,	6	410	5
R6	SGI,Origin 3200 IRIX,	16	410	5
R7	SGI,Origin 3200,IRIX	6	410	4
R8	Intel,Pentium/VC820,Linux	2	380	1
R9	SGI,Origin 3200,IRIX	4	410	6

**User Model.** Each User submits a batch job constrained by deadline and budget. Each task is described by length. Length, ranging from 0 to  $300 \times 377MI$ , is generated by Gridsim randomly.

The budget and deadline of each job are decided by its characters and performance of grid resources. They are calculated as follows [13]:

$$\text{Deadline: } \Gamma = T_{\min} + D_{factor} \cdot (T_{\max} - T_{\min}) \quad (7)$$

Where,  $T_{\min}$  is executing time when all the tasks are parallel executed with fastest resources and in highest priority;  $T_{\max}$  is executing time when all the tasks are serially executed with slowest resources;  $D_{factor}$  is adjusting factor. If  $D_{factor} < 0$ , we consider the batch job can't be finished.

$$\text{Budget: } B = C_{\min} + B_{factor} \cdot (C_{\max} - C_{\min}) \quad (8)$$

Where  $C_{\min}$  is the cost where all the tasks execute in the cheapest resources, and  $C_{\max}$  is the cost where all the tasks execute in the most expensive resources.  $B_{factor}$  is the adjust factor of the cost. If  $B_{factor} < 0$ , we consider the batch job can't be finished.

**Experiments and Result.** Given the grid model and the user model above described, we conducted experiments on the basis of the reverse auction-based grid resources allocation simulating system that we developed, and did experiments under the same condition by using the Nimrod-G simulator [13] with *cost-optimization* policy. The Nimrod-G simulator is based on the commodity market and issued in the Gridsim package.

We did many experiments with different combinations of User number, budget factor and time factor. In every experiment, User enters into the global grid in random time ranging from 0 to 1000 time units and begins to do his experiment. Each User has a batch job consisting of 40~80 tasks. The task's length, job's deadline and budget are described in user model. The experiments were conducted with User number taken as 1, 5, 15, 20, 40, 60, 80, or 100 each time.

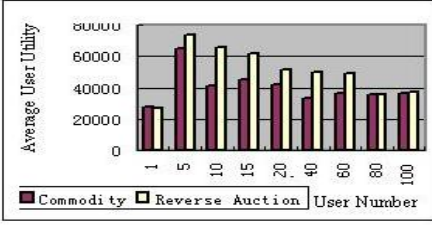


Fig. 3. Average User Utility

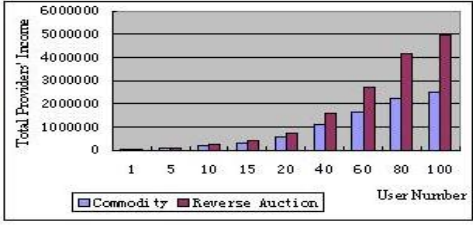


Fig. 4. Total Providers' Income

Fig.3 shows average user utility. In reverse-auction market, if the number of users is above 5, as the number of user increases and the competition is fiercer, the price of computational power will increase, and the average of user utility decreases. Since the price in the commodity market is constant, the average of user utility fluctuates within 30,000 and 45,000 if the number of user is above 10. Experiments show that reverse auction-based resources allocation mechanism leads to a better user utility.

Fig.4 shows total income of Providers in two resources allocation methods. The total income in the reverse-auction market is higher than that in commodity-based market because Providers increase resources price when competition is fiercer.

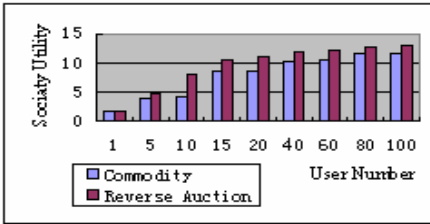


Fig. 5. Society Utility

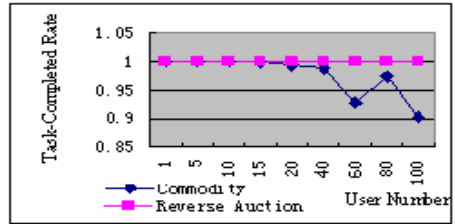


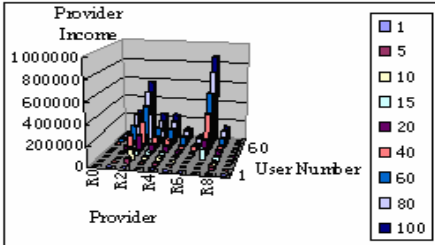
Fig. 6. Task-Completed Rate

Society utility represents justice of resources allocation and load-balance, and is shown in Fig.5. Resources allocation model's society utility of reverse auction-based is higher than that of commodity-based when user number is larger. On one hand, competition is severe when the user number increases, and results in resources' price rise, so Providers gain more profit. On the other hand, each Provider changes its resources price dynamically according to the relationship of supply and demand in order to win an auction; this also results in that the allocation of task tends to balance. However, in the commodity market, the price of resources is constant and the user selects low price's resources, this makes the job is allocation to some low price resources and result in workload and income being unbalance.

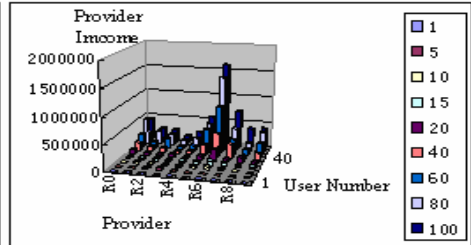
In Fig.6, the rate of completed tasks is defined as the number of tasks being finished by the global grid system divided by the total number of all tasks. It represents the effect of resources allocation. All jobs were finished in reverse auction-based experiments. The rate of completed task decreases in commodity-based experiments as the number of user/task increases. The reasons are as follows: First, as described

above, the reverse auction based resources allocation method allocate tasks to all providers in balance, therefore all resources are fully used. But the commodity based resources allocation allocates the task to the low price resources, therefore some resources were abused and some others idled. Second, the Nimrod-G simulator is a centralizing allocation method, and centralizing method does not work well as the user number increases. Contrarily, our method is distributed and the price is used as heuristic message for task schedule, so it works well even in the case of large number of users.

The distribution of Provider income was shown in **Fig.7** and **Fig.8**. In **Fig. 7**, Users select cheap resources and the price in commodity market remains constant for a long time, which makes the cheap resources be abused, and expensive resources be left unused. In **Fig. 8**, because the Provider changes the resources price dynamically according to supply-demand relationship in reverse auction-based market, each Provider is allocated task in balance according to his total power, he gains relative balance income according to his power.



**Fig. 7.** Commodity Market



**Fig. 8.** Reverse Auction Market

## 5 Related Works

Some researches on the market based resources allocation have been conducted. Nimrod-G is a famous task scheduler based on the commodity market mechanism. As the comparisons we have made in Section 4.3, our reserved auction-based resources allocation is better than Nimrod-G in the user utility, society utility, load balance, job-completed rate, etc. Although auction protocol was discussed in [4], no details were given, and it was not used in a real scheduler or scheduler simulator. The communication demand or complexity of auction protocols was investigated in Grid environments in [8], but didn't combine with the resource allocation.

Grid Federate [5] is another market based resources allocation model. It can allocate task more balanced than Nimrod-G. It is based on commodity market. But Providers in the Grid Federate are cooperative more than competitive. This is different from our complete competitive market, which is fit to the real world.

Market-based schedule mechanism was discussed in [14,15,16,17] for distributed computing. They use the price as a heuristic message for task schedule, resulting in better performance in the load-balance. However, they didn't take the user utility into account, and didn't support the Qos requirement. In our allocation mechanism, we support the user Qos requirement, and take the user utility into account.

## 6 Conclusions

Resources management and allocation is a key and challenging technology in grid system. We proposed a reverse auction-based grid resources allocation mechanism; some price algorithms for Provider and User were designed. A reverse auction-based resources allocation simulator was developed on the basis of Gridsim. Experiments showed that the reverse auction-based resources allocation mechanism is better than Nimrod-G in the user utility, society utility, load balance, job-completed rate, etc. The further work is to put some artificial intelligence into the auction algorithm and pricing algorithm to improve the allocation result and performance, and apply this mechanism to a real grid system.

## References

1. Rajkumar Buyya. Economic-based Distributed Resource Management and Scheduling for Grid Computing, Ph.D. Thesis, Monash University, Melbourne, Australia, April 12, 2002.
2. Fran Berman, Anthony J.G. Hey, Geoffrey Fox. Grid Computing: Making The Global Infrastructure a Reality. John Wiley and Sons Ltd. 2003.
3. Rajkumar Buyya and Srikumar Venugopal, The Gridbus Toolkit for Service Oriented Grid and Utility Computing: An Overview and Status Report, CoRR, cs. DC/0404027(2004)
4. Rajkumar Buyya, David Abramson, and Srikumar Venugopal, The Grid Economy, PROCEEDINGS OF THE IEEE, VOL. 93, NO. 3, MARCH 2005.
5. Rajiv Ranjan, Aaron Harwood and Rajkumar Buyya, Grid Federation: An Economy Based Distributed Resource Management System for large-scale Resource Coupling. Technical Report, GRIDS-TR-2004-10, Grid Computing and Distributed Systems Laboratory, University of Melbourne, Australia, Dec 7, 2004.
6. Srikumar Venugopal and Rajkumar Buyya, An Economy-based Algorithm for Scheduling Data-Intensive Applications on Global Grids, Technical Report, GRIDS-TR-2004-11, Grid Computing and Distributed Systems Laboratory, University of Melbourne, Australia, Dec 8, 2004.
7. Chee Shin Yeo and Rajkumar Buyya, A taxonomy of market-based resource management systems for utility-driven cluster computing, Software: Practice and Experience, Wiley Press, USA. (September 2005)
8. Marcos Dias de Assuncao and Rajkumar Buyya, An Evaluation of Communication Demand of Auction Protocols in Grid Environments, Proceedings of the 3rd International Workshop on Grid Economics & Business (GECON 2006), May 16, 2006, Singapore.
9. Giorgos Cheliotis, Chris Kenyon, and Rajkumar Buyya, Grid Economics: 10 Lessons from Finance, Peer-to-Peer Computing: Evolution of a Disruptive Technology, Ramesh Subramanian and Brian Goodman (editors), Idea Group Publisher, Hershey, PA, USA.
10. Rajkumar Buyya, Manzur Murshed, David Abramson, and Srikumar Venugopal, Scheduling Parameter Sweep Applications on Global Grids: A Deadline and Budget Constrained Cost-Time Optimisation Algorithm, CoRR cs. DC/0203020: (2002).
11. Zhengyou LIANG, Ling ZHANG, Shoubin Dong, and Wenguo WEI. Charging and Accounting for Grid Computing System. GCC2003, LNCS 3033/2004.
12. Chunming Chen, Muthucumaru Maheswaran, and Michel Toulouse. Supporting co-allocation in an auctioning-based resource allocator for grid systems. In Proceedings of the International Parallel and Distributed Processing Symposium (IPDPS 2002): 306.

13. R. Buyya and M. Murshed. GridSim: A toolkit for the modeling and simulation of distributed resource management and scheduling for grid computing. *J. Concurrency and Computation: Practice-Experience (CCPE)*, 2002,14: 1175–1220.
14. J. Kurose and R. Simha. A microeconomic approach to optimal resource allocation in distributed computer systems. *IEEE Trans. on Computers*, 38(5), 1989:705--717.
15. CA Waldspurger, T. Hogg, B. Huberman, J. Kephart, and S. Stornetta, Spawn: A distributed computational ecology, *IEEE Trans. on Software Engg.*, 18(2), February, 1992.
16. Kazuhiro Kuwabara, Toru Ishida, Yoshiyasu Nishibe and Tatsuya Suda, An Equilibratory Market-Based Approach for Distributed Resource Allocation and Its Applications to Communication Network Control, Scott H. Clearwater Ed., *Market-Based Control: A Paradigm for Distributed Resource Allocation*, World Scientific Publishing, 1996.
17. M. A. Gibney and N. R. Jennings. Dynamic resource allocation by market-based routing in telecommunications networks. In 2nd Int. Workshop on Multi-Agent Systems and Telecommunications, 1998: 102--117.

# Data Grid System Based on Agent for Interoperability of Distributed Data

Youn-Gyou Kook, Gye-Dong Jung, and Young-Keun Choi

Dept of Computer Science Kwangwoon University  
Seoul, 139-701, Korea  
{ykkook, kdjung, ykchoi}@cs.kw.ac.kr

**Abstract.** This paper presents data grid system based on agent for an efficient interoperability of distributed data. Purpose of the data grid system is an efficient interoperability through sharing and exchanging data among distributed databases. Interoperability of distributed data is a basic condition for cooperation of existing legacy systems. More nodes for cooperation, better for distributed environment. On the other hand, we have to overcome heterogeneous characteristics of data and physical problems of network. Therefore, we use agent system which is independent, autonomous and mobile in distributed platform for solving this problem. As mobile Agent interacts asynchronously, it could lessen network traffic and support workload and distributed processing of services so that it is suitable for data grid system. Also we use XMDR(eXtended Metadata Registry) in ISO/IEC 11179 for sharing and exchanging data and overcome the heterogeneity of data. Therefore we improve availability and transparency of distributed data based on data grid middleware using proposed mobile agent and could construct data grid system with low cost.

## 1 Introduction

The recent issue about providing the latest information for minimizing latency during process execution and management of important work in real-time enterprise that is appeared by necessity of new strategy for corresponding fast change speed of enterprise environment is emphasized. Data interoperability in real-time enterprise environment is a field of data grid for utility of efficient distributed data and has lively researched. Data grid is a technology for efficiently using distributed data through sharing and exchanging data among distributed database environment. Most researches about grid focuses on the application of place where the data are stored. However recently, it is more important to connect legacy systems and databases which are independently operating. Oracle, providing database administrator system, and IBM have researched data grid based on database. This paper provides middleware for supporting data grid system whose objectivity is interoperability of efficient distributed data in distributed database. As interoperability of distributed data is a basic condition for cooperating existing legacy systems, availability of data, transparency and physical problem of participating nodes to the cooperation have to be resolved. The proposed data grid middleware uses mobile agent and XMDR for solving

these problems. Mobile agent can use distributed data at anywhere, lessen network traffic, support distributed processing of workload and service and solve physical problems because it is independent from platform, autonomous, and mobile. We also use XMDR for solving transparency of distributed data. XMDR is extended version of MDR and ontology. MDR is a standard for sharing and exchanging information at ISO/IEC 11179[11]. Ontology is for efficient usage of information. The design of XMDR is suitably for interoperability of distributed heterogeneous data using XML which is web standard document. We can construct database grid for integrating legacy systems with low cost and improve availability and transparency of data through dynamic service for data access, integration and management which are provided by proposed data grid middleware. Section 2 presents data grid for data interoperability, and section 3 designs XMDR for data transparency. Section 4 presents data grid system and section 5 describes grid middleware. Section 6 presents services provided by the system. Section 7 shows implement result and section 8 is conclusion.

## 2 Related Work

Data grid is a technology for supporting data interoperability by efficient use of distributed storages which are networked. Recently databases of legacy systems need to connect each others. Grid technology has been researched for it. There is only focus on the application about the place where data could store in the research trend of data grid. There is integration of storage devices, information retrieval, data naming for representing distributed data location, replication management and consistence maintenance of data, sharing and protection of data and data access management in the research area. The researches for interoperability of data and access management of distributed database are lively progressing based on the data grid research, especially in real time enterprise. The paradigm of real time enterprise means providing real time information and integration at virtual space not physical space. Providing real time information is lessening the information providing time to minute units by integrating systems.

Grid-DBMS[6] proposes specification which present framework for data management in Grid and is a research field in data grid. Especially, it emphasizes openness, economical efficiency, and availability of grid, and proposes the system for dynamic management to distributed data. Just concerning a part of data access and management of distributed system centered, it overlooks the data heterogeneity for data interoperability. For interoperability between heterogeneous systems for data exchange based on XML, X-MAP[4], which connects automatically among meanings of schema element by defining different-sounded-synonym, considers only data heterogeneity and there are no comment about system heterogeneity. Grid solution provided by Oracle[9] emphasizes middleware offering for interoperability of data by constructing grid based on relation database, but not interoperability of heterogeneous databases. This paper presents data grid middleware for interconnecting distributed databases and describes interoperability of distributed data based on definition and design of XMDR.

### 3 Design of XMDR(Extended Metadata Registry)

XMDR, as an extended concept of ontology for efficient use of information and MDR which is a standard for proposed sharing and exchanging information, have been researched. Especially metadata for managing share data in 3 part of ISO/IEC 11179 is proposed and meta-model offers the standard and guide for construction of data elements shared between systems, users in distributed environment and semantic contents.

This paper constructs ontology for efficient use and constitutes MDR based on database of legacy systems which constitutes data grid. Standard element of information for cooperation in data grid is constructed through standard ontology and location ontology for efficiency. Designing XMDR based on MDR and those two ontologies, XMDR is constructed with XML for heterogeneous computing environment. Also all data movement for cooperation in data grid use XML based on data properties presented in standard ontology. Therefore, one can participate the cooperation without modifying semantic of metadata or structure modification of legacy systems constructed data grid. XMDR which is proposed in this paper defines location ontology, standard ontology and MDR. First of all, location ontology is constructed based on database of legacy system constructed data grid. Location ontology has the properties which are database information, LOC\_URL which is location information and LOCID for identifying location.

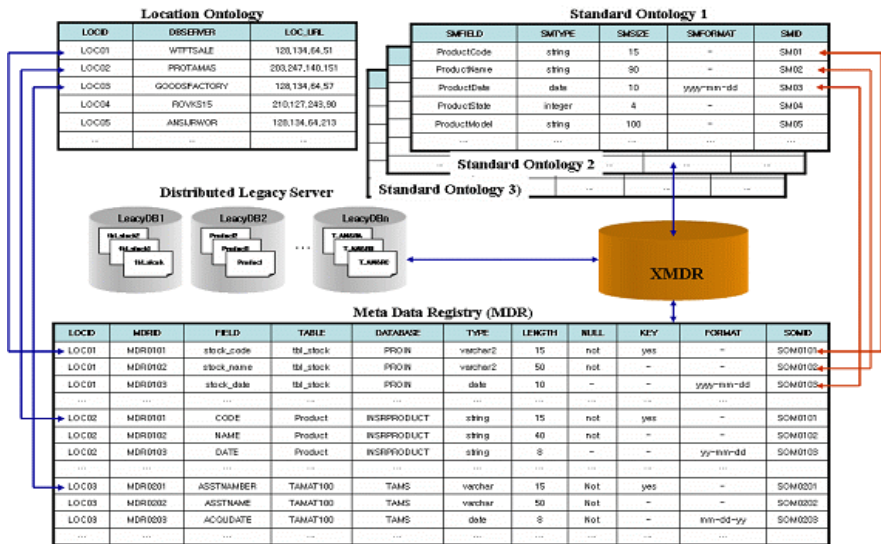


Fig. 1. MDR and ontology constructing process for interoperability of distributed data

Standard ontology, configuring standard element for sharing and exchanging data, is constructed. Standard ontology is constituted by group for cooperation and presents semantic contents of metadata and standard for structure of data element. We present standard glossary, data type, size and presenting method for overcoming data



heterogeneity used in legacy system. Especially, standard glossary uses obvious glossary for solving semantic heterogeneity of data. Metadata registry is constructed for sharing and exchanging information of distributed data based on the location and standard ontology. MDR follows property specification of proposed data in ISO/IEC 11179-3. The basic properties of data are identity, definition, relation and present property and shown below.

- Identity property: Property for identifying data element(MDRID)
- Definition property: Property that contains semantic of data element(FIELD)
- Relation property: Property that presents relation of data element and entity or data elements(TABLE, DATABASE)
- Present property: Property about how to present data element(TYPE, LENGTH, NULL, KEY, FORMAT)

MDRID is for identifying shared data and used when fields defined in database in legacy system are identified. FIELD means the defined field in database schema of each systems and needs database and table information as an relation property and there are TABLE and DATABASE for constructing various data into one XMDR. Also present property presents data type and size, null value check, key value and present method. Efficient data sharing and exchanging can be performed by using XMDR constructed based on defined MDR and ontology. XMDR can guarantee autonomy and independency of distributed data because it situates in legacy systems and is used for data access. It can be used without modifying reconstruction of data schema of legacy system for constructing data grid. Figure 2 shows XMDR described by XML for participating legacy system in data grid to use.

```

<?xml version="1.0" encoding="euc-kr" ?>
<!DOCTYPE XMDR SYSTEM "XMDR_schema.xml">
<XMDR>
  <SOMID som="1" db="yes">
    <LOCATION id="1">128.134.64.51</LOCATION>
    <DATABASE>PRODIN</DATABASE>
    <TABLE>tbl_stock</TABLE>
    <MDRID mid="MDR0101">
      <FIELD type="varchar" length="15" null="no" key="yes" format="no">stock_code</FIELD>
      <SR type="string" size="15" format="no">ProductCode</SR>
    </MDRID>
    <MDRID mid="MDR0102">
      <FIELD type="varchar" length="50" null="no" key="no" format="no">stock_name</FIELD>
      <SR type="string" size="90" format="no">ProductName</SR>
    </MDRID>
    <MDRID mid="MDR0103">
      <FIELD type="datetime" length="0" null="no" key="no" format="yyyy-mm-dd">stock_date</FIELD>
      <SR type="date" size="10" format="yyyy-mm-dd">ProductDate</SR>
    </MDRID>
    <MDRID mid="MDR0104">
      <FIELD type="int" length="4" null="yes" key="no" format="no">stock_state</FIELD>
      <SR type="integer" size="4" format="no">ProductState</SR>
    </MDRID>
    .....
    .....
  </SOMID>
  <SOMID som="2" db="yes">
    <LOCATION id="1">128.134.64.51</LOCATION>
    <DATABASE>PRODIN</DATABASE>
    <TABLE>tbl_wacc</TABLE>
    <MDRID mid="MDR0101">
      <FIELD type="varchar" length="15" null="no" key="yes" format="no">wacc_code</FIELD>
      <SR type="string" size="15" format="no">WaccProductCode</SR>
    </MDRID>
    <MDRID mid="MDR0102">
      <FIELD type="varchar" length="50" null="no" key="no" format="no">wacc_name</FIELD>
      <SR type="string" size="90" format="no">WaccProductName</SR>
    </MDRID>
    .....
    .....
  </SOMID>
</XMDR>

```

Fig. 2. XMDR design for interoperability of distributed data

XMDR is a tool for standard conversion about data access of application program or data movement of legacy system. When performing necessary job for data gathering or event of data located in each area, standard and location conversion is performed using XMDR. As you see in Figure 2, property of SOMID element is

determined by standard ontology and configured URL of LOCATION element which presents location ontology. Also database information and table information are defined in DATABASE and TABLE elements and information about each data schema is defined in MDRID element. Schema of legacy system is defined in FIELD element and property and schema of standard ontology is defined at SM element. When existing multiple data schema for the cooperation, multi construction of standard ontology is needed.

## 4 Data Grid System Using Agent

### 4.1 System Outline

Data grid community constructed based on data grid middleware using proposed agent system in this paper is shown in Figure 3. Participating nodes for cooperation in data grid community are constructed with legacy systems. Cooperation job, such as file sharing, distributed computing, data sharing and exchanging and information retrieval, is performed by its purpose.

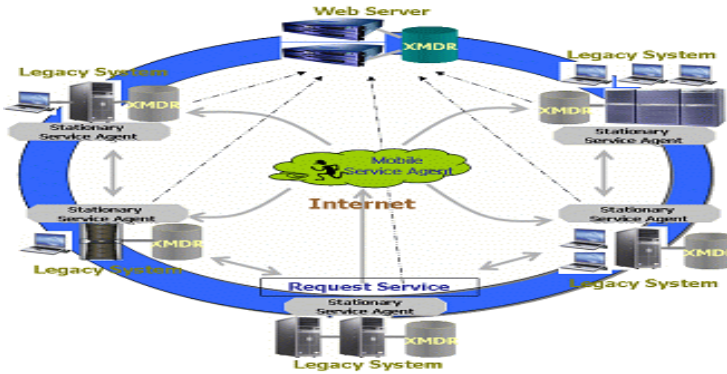


Fig. 3. Data Grid Community Outline

Nodes constituted the data grid community as legacy systems have to take data grid middleware. Just need to install agent system which is data grid middleware, so that there is no modification by constituting community. Agent system has two agents for data grid service; stationary service agent in local and mobile service agent for performing service in other node after moving.

If services are requested in the web or application using data grid service, it requests necessary job to each nodes through agent system as in Figure 3. Job processing is requested using two service agents in the requesting node. Stationary service agent exists in every agent systems, performs requested job and return the result and mobile service agent has a service which other nodes do not have so that it migrates to others and executes its job. First, jobs needing to synchronously connect with other each nodes request executable jobs to stationary service agent and jobs needing to

asynchronously connect process with mobile service agent. Mobile service agent is migrated by migration path from agent administrator of requesting node.

### 4.2 System Design

The proposed data grid middleware in this paper constitutes with agent system and grid mechanism. The middleware can perform client and server role at the same time and needs resource management and allocation for processing jobs and resource access of node. For these functions we need follow required mechanism.

#### 4.2.1 Community Mechanism

Community Mechanism is based on node priority policy for job scheduling which requests to distributed nodes and node discovery policy participating in grid community. Node discovery uses directory server of community and directory server processes authentication of node participating in the community. Also community is limited to be extended because it is constituted with objective of cooperative job.

Node priority policy decides job priority to distributed nodes for requesting jobs and let the job perform efficiently. Job priority of node is decided with three methods. Firstly, it is decided with participating order of nodes to the community. This method decides node priority by lists based on time-stamp of node participated in the community. Secondly, the decision is made by a user who wants to request job selects a node. This method is based on user's request. Finally, it is decided by information of node that wants to request a job. Information of distributed node monitors performance of node by directory server.

#### 4.2.2 Agent Mechanism

Job in data grid community is performed by service agent and service agents are divided into two part. Stationary service returns the result after processing a job by job request that need to connect synchronously in other nodes, but mobile service performs after migrating to other nodes. Migration of mobile service agent affects to job execution time.

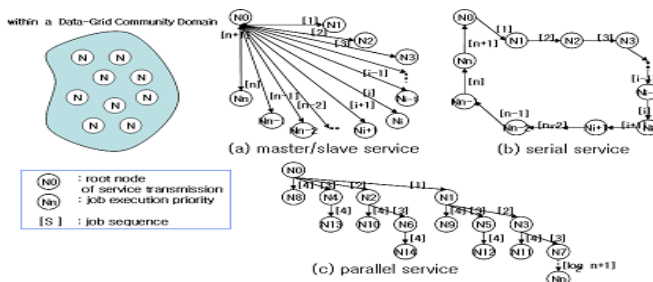


Fig. 4. Job process order of service agent

Migration path is decided by transmission method of mobile service agent. Migration path is decided by transmission method of node that participate in the community; master/slave, linear and parallel transmission method. Migration path of nodes

for transmitting mobile service follows decided order by fore-described node priority policy. Figure 4 shows migration mechanism for transmitting mobile service agent from one data grid community.

Migration method of mobile service agent is selected by job characters and scope of job process rather than job process time. We use master/slave migration method shown in (a) of Figure 4 for the jobs that have many data and low process scope or when requested migration service is transmitted to only one node. Also after transmitting a job, we use linear transmission method shown in (b) of Figure 4 when the migrating data is few after job processing or when needing to just job processing result. Finally, we use parallel transmission migration method shown in (c) of Figure 4 when job process scope is bigger or when faster job processing is required.

### 4.2.3 Agent Life Cycle

Extension of nodes in data grid environment can improve job processing. If objective of community has conditional constraints, the number of nodes is not infinitely increased. We have to concern this situation when configuring life-cycle of service agent. Life-cycle of service agent controls with TTL(Time to Live) value that limits number of job requesting nodes. If TTL is zero(0), requesting to all nodes participating in data grid environment is possible and if it has positive number, requesting to nodes a jobs is possible as the positive number is. Especially, in the case of migrating with itinerary of mobile service agent, the agent can be controlled efficiently by configuring TTL value of agent. Infinite migration of mobile service agent cause rather bad processing result so that configuring TTL by job request might be good for the result.

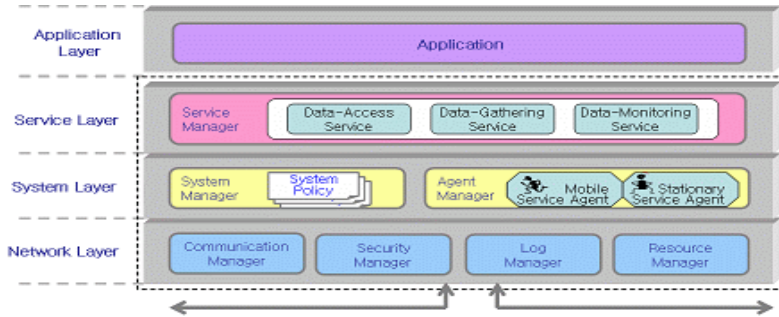
### 4.2.4 Fault Tolerance

The faults during job processing are node, system and network fault. Fault tolerance policy must be necessary for distributed environment and it offers reliability and robustness of the job. Fault tolerance mechanism uses two techniques by whether job is processing or not. The fault that is occurred when trying to connecting for requesting a job uses re-order technique and backward recovery technique. The fault that is occurred when processing a job constitutes fault tolerance mechanism using checkpoint technique.

## 5 System Architecture

Data grid middleware is comprised of 4 layers that contains management module and application layer for constructing data grid. Management module participates to data grid community and there are two layers; network layer for forming grid network and system layer that construct data grid middleware. And service layer is constructed for providing data grid service. Figure 5 shows architecture of data grid middleware.

Fundamental functions and descriptions of management module of each layers constructing data grid middleware are shown below:



**Fig. 5.** Architecture of Agent System

- Communication management: a module for managing communication channel formation. It decides whether job is requested or not by checking status information of requesting node. Communication manager manages communication channel of transmitted/received job request in wait state and provides management module for migration guarantee of migrating service agent.
- Resource management: Node resources are processor, memory, network and data related resources. It monitors resource status of node. Resource status information of node is transmitted to directory server by communication manager.
- Security management: Data grid middleware based on proposed agent is coded with only java so that it uses security module of java. Authentication of node is identified in the directory server and it authenticates transmission of agent. Agent transmitting to a node processes the job only when allowed access of system.
- System management: Agent system is middleware installed in legacy system participating data grid community for cooperation work. The middleware is constructed based on P2P[8] and a agent system that performs both client and server. Each node transmits node information to directory server for participating community.
- Agent management: a module that manages service agent by job request and manages stationary service agent which transmits only job request to agent and mobile service agent which performs service after migrating to other node. It manages job process of service agent formed by service manager, life cycle, and service agent and job priority.
- Log management: manages the record of all events that use data grid network. It records events such as status of node, agent transmission/reception, job request and so on.
- Service management: manages service that forms processing job in data grid community. The service is a kind of job that is processed after transmitting to each node of community and performs various jobs.

The services provided in this paper access of distributed data and management services. These provide Data Access Service(DAS), Data Gathering Service(DGS) and Data Monitoring Service(DMS) because the objective is access and management of distributed data as cooperation work of data grid community. Service manager

comprises of fundamental modules for providing each services. Description of the services is in next chapter. Application layer is for using data grid service. Various applications are utilized based on the services provided from community.

## 6 Services Provided in Data Grid Communication

This chapter describes services provided in data grid communication. The services provided in the community which is formed based on proposed data grid middleware access and manage distributed data. Work for data management is accessing, collecting and monitoring data. This needs to be provided with interface of relation or non-relation data source. The interface for accessing data source existing in each legacy system has to be provided when agent system, which is grid middleware, is configured.

Data access and management service is shown below:

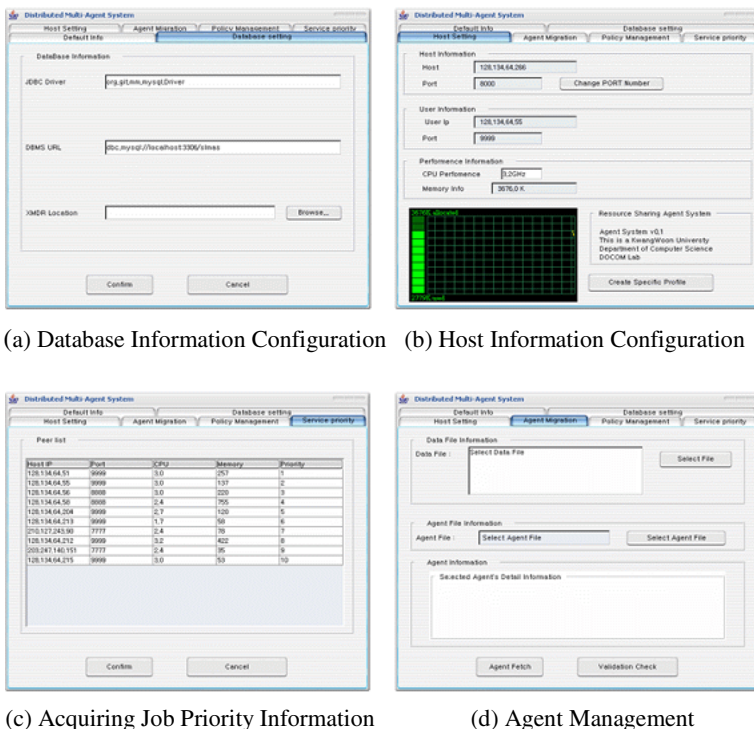
1. **Data Access Service(DAS):** This service can access data source located in distributed nodes. Application layer, located over service layer, can access the distributed data source using this service. Data Access Service provides necessary module for managing data source in the grid community. Accessing data source can be possible without concerning real location and heterogeneity of data source. Distributed node has to configure the location of data source and uses XMDR for overcoming heterogeneity of data source.
2. **Data Gathering Service:** This service can gather necessary information accessing data source located in distributed node. Data source set of distributed node is shown as one logical data source. In fact there are several distributed data sources but users feel as if gathering from one data source. Data Gathering Service gathers result data that solves heterogeneity of presenting, data type and semantic of data. For interoperability of distributed data, it can gather more efficient data.
3. **Data Monitoring Service:** This service monitors community data for acquiring useful information for decision making. It transmits node information that is monitored of data change to application which uses data monitoring service and gathers fast data of data change.

## 7 Implementation and Evaluation

Data grid that is constructed for data interoperability of distributed databases in real-time enterprise environment is comprised of legacy systems so that independence and autonomy of each legacy systems have to guarantee. Therefore, middleware for data grid construction uses agent system and XMDR for data sharing and exchanging standard. Agent system, which is data grid middleware, is platform-independent and developed with Java for supporting asynchronous communication of nodes. Web-based real-time monitoring is also implemented concerning with management efficiency of agent system.

## 7.1 System Implementation

Agent system, which is data grid middleware, is in Figure 6. Host and database information of legacy system have to be configured in agent system. Agent system is implemented with java so that interface that is for accessing databases in legacy systems, JDBC and URL, is configured and this is in (a). For making data grid monitoring in web configure information of directory server for transmitting host information to directory server as shown in (B). Host information transmitted to directory server includes resource information of agent system. Itinerary to execution order of cooperation work of legacy system is acquired by directory server and acquired screen is shown in (c). Job priority is decided based on legacy system information, manager configuration and time-stamp. Service execution by system priority is executed by service agent. There are two types of agents; stationary agent that is common processing job in legacy system and mobile service agent that requests job and it is created when it is needed. Selection of data file by service agent and agent transmission is shown in (d).



**Fig. 6.** Policy configuration of Agent System

If the agent system runs in legacy system formed data grid, it transmits information of agent system to directory server. Directory server manages XMDR that is defined for interoperability of distributed data and information of agent system. Figure 7

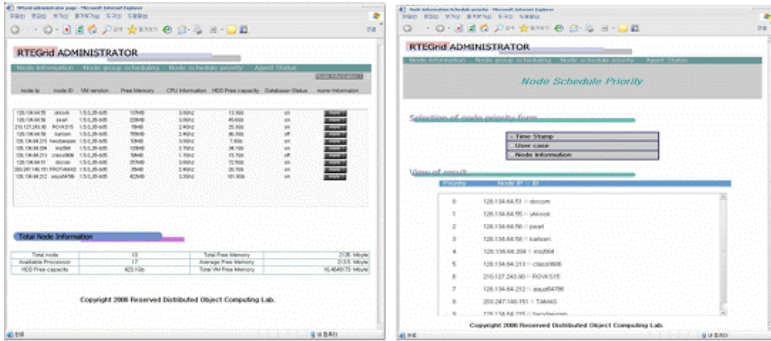


Fig. 7. Web Management of agent system

shows how to decide job priority and monitor information of agent system. It also manages information about each system and defined XMDR.

7.2 Evaluation

The researches for interoperability of distributed data have been lively progressing. Especially, these are focused on heterogeneity of data. BizTalk[12] of Microsoft uses Mapper and helps data exchange make direct mapping between schema. X-MAP[4] system connects among semantic of schema so that it exchanges data and it is proposed for data interoperability of multi heterogeneous systems. Grid-DBMS[6] also presents a framework for dynamic data management in grid environment and there is Oracle grid[9] for information management in real-time enterprise environment. As you see in these researches, the researches for interoperability of distributed databases have to overcome heterogeneity of system and data.

We compare the proposed system with other systems and there are four kinds of comparison elements. First, whether it support MDR(ISO/IEC 11179) that is a standard of data integration, data auto-exchange after designing schema, synchronization of data or not and system independency concerning with heterogeneous system in distributed environment or not.

Table 1. Comparison with data exchange system

	BizTalk	X-MAP	DBMS-Grid	Oracle-Grid	proposed system
MDR(ISO/IEC 11179)	Non supported	Non Supported	Non supported	Non supported	Supported
Data exchange automation	Supported	partly supported	supported	partly supported	Supported
Data Synchronization	partly supported	Non supported	partly supported	partly supported	partly supported
System Independence	Non supported	Supported	Supported	Supported	Supported



We test the performance of service agent with migration scheme of service agent. The number of nodes is from 5 to 40 systems. Even it is the simple operation job; the entire execution time is increased linearly as the number of nodes is increased in Master/Slave service and serial service as shown at fig. 8. Also, the parallel service shows almost flat line, because service agent reduces migration time.

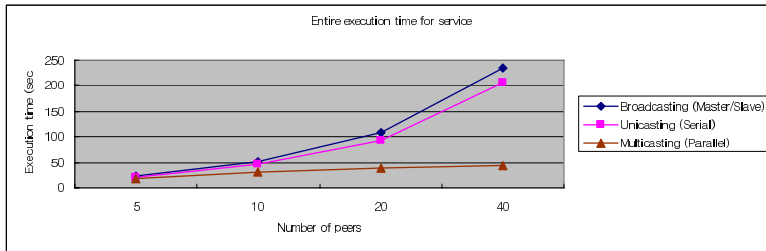


Fig. 8. Entire Execution Time for Service Agent

## 8 Conclusion

The researches of data grid based on database for sharing real-time information and knowledge of organization in real-time enterprise environment have been lively researching and especially data grid construction for cooperation of existing legacy systems is more emphasized. This paper presents data grid system based on XMDR for interoperability of distributed data in those real-time enterprise environments.

XMDR, as a integrated standard for interoperability of data, can solve data heterogeneity and easily and efficiently access data reducing unnecessary procedures by ontology. The system is constructed with data grid for cooperation of existing legacy systems so that it is easy to access distributed data. Agent system used as data grid middleware that interacts asynchronously lessens network traffic; supports distributed processing of service and load balance and can provide service that is data accessible of user-centric except existing services.

Data interoperability using data grid middleware based on proposed XMDR in this paper makes fast and efficient access and share of distributed data possible. Future work is the additional research of metadata for data interoperability among domains and extension of data grid community and service providing with web service for utilization of data grid.

## Acknowledgement

The present Research has been conducted by the Research Grant of KwangWoon University in 2006.

## References

- [1] Balazs Goldschmidt, Zoltan Laszlo, "Mobile Agents in a Distributed Heterogeneous Database System", *Parallel, Distributed and Network-based Processing*, IEEE, pp.123- 128, Jan., 2002.
- [2] David Barkai "Peer-to-Peer Computing Technologies for Sharing and Collaborating on the Net", Intel Press, pp.181-191, 2002.
- [3] D. Thain, T. Tannenbaum, and M. Livny, "Distributed Computing in Practice: the Condor Experience", *Concurrency and Computation: Practice and Experience*, Vol. 17, Issue 2-4, pp.323-356, Feb. 2005
- [4] David Wang, "Automated Semantic Correlation between Multiple Schema Information Exchange", M.I.T, MM, May, 2000.
- [5] E. Bertino and B. Catania, "Integrating XML and databases", *Internet Computing*, IEEE, Vol.5, No.4, pp.84- 88, July, 2001.
- [6] G Aloisio, M Cafaro, S Fiore, M Mirto, "The Grid-DBMS: Towards Dynamic Data Management in Grid Environments", *ITCC'05*, IEEE Vol. 2, pp199-204, 2005
- [7] Giacomo Cabri, Letizia Leonardi, Franco Zambonilli, "Mobile-Agent Coordination Models for Internet Applications", *Computer*, IEEE, Vol.33, Issue 2, pp. 82-89, Feb., 2000.
- [8] Karl Aberer, etc 6, "P-Grid: A Self-organizing Structured P2P System", *SIGMOD Record*, 2003
- [9] Matthew Hart, Scott Jesse, "Oracle Database 10g: High Availablility with RAC, Flashback & Data Guard", Oracle Press, 2005
- [10] Stavros Papastavrou, George Samaras, Evaggelia Pitoura, "Mobile Agent for World Wide Web Distributed Database Access", *Knowledge and Data Engineering*, IEEE Transactions on, Vol.12, Issue 5, pp.802-820, Sept.-Oct., 2000.
- [11] ISO/IEC IS 11179, "Information technology -Specification and standardization of data elements", 2003.
- [12] Microsoft BizTalk, "<http://www.microsoft.com/biztalk/>"

# A Layered Semantics for Mobile Computation\*

Jianghua Lv<sup>1,2</sup>, Shilong Ma<sup>1</sup>, Jing Pan<sup>3</sup>, and Li Ma<sup>1</sup>

<sup>1</sup>National Lab of Software Development Environment, Beijing University of Aeronautics and Astronautics, Beijing, China, 100083

{jhlv, slma, mali}@nlsde.buaa.edu.cn

<sup>2</sup>College of Computer Science and Technology, Jilin University, Changchun, China, 130012

<sup>3</sup>School of Management, University of Science and Technology Beijing, Beijing, China, 100083

panjing@manage.ustb.edu.cn

**Abstract.** Since mobile computation involves variants information and different behaviors, it is very complicated in mobile computation systems. In order to analysis mobile computation and understand the essential of mobile computing, in the paper we develop a denotational semantics for mobile computation. Here we take CLAIM, a computational language for autonomous, intelligent and mobile agents, as our object language because it characterizes the essential ingredients in mobile computation. After abstracting the syntax construction of CLAIM, we give its semantics description. This is achieved by structuring the semantics in layers working at three different levels: internal ambient, ambients and programs based on the basic concept: ambient. For each of these three levels, their semantics are defined in detail as well as the relationship between levels. Through our approach, we can also obtain an explicit model of behaviors in mobile computation.

## 1 Introduction

Nowadays, with the development of world-wide-web, mobile computation has become a main focus of research. The geographic distribution of the Web naturally calls for mobility of computation, as a feasible way to utilize services available as possible as we can. Especially now most recent advances in networking and language technology are gained, how to effectively use and share the internet services as the main character in mobile computation are technologically realizable. To achieve the goal, however, we need to analysis the behaviors in a mobile computation system and understand how it works.

Mobile ambient, a calculus for mobile computation, is first introduced by Cardelli and Gordon in 1998 as a model to describe the movement of processes and devices, including movement through administrative domains [1, 2]. They describe all these aspects of mobility within a single framework that encompasses mobile agents, the ambients where agents interact and the mobility of the ambients themselves. In mobile

---

\* Supported by the National 973 Project under the grant number G1999032701 and China Postdoctoral Science Foundation.

ambient, each ambient is a bounded place in which computation happens, the computation is called service or processes, so an ambient can be move as a whole and the service in an ambient can be moved out from an ambient or enter into another ambient, any ambient may be nested within other ambients or may contain sub-ambient, these movements are achieved through the use of *capabilities*, which let the structure of an ambient involve moving actions and direct the movement of the ambient.

However the distances between the ambient calculus and practical systems are still large, in order to help the designer to reduce the gap between the design and the implementation phases, agent-oriented programming languages need to be proposed, at the same time these languages should also meet the requirements of mobile computation which becomes popular due to the recent developments in the mobile code paradigm. To satisfy these demands, many agent-oriented languages spring up, for example, AGENT-0 [3], PLACA [4], VIVA [5], 3APL [6] and so on, There are also many languages oriented on representing agents' or processes' mobility and concurrency. Among these, CLAIM [7, 8], a computational language for autonomous, intelligent and mobile agents, allows to design Multi-Agent Systems (MAS) that support both stationary and mobile agents and at same time it characterizes the essential ingredients in mobile computation, therefore we chose CLAIM as our description object.

As we know, in the design of a programming language, a formal study of its semantics can be of considerable advantage [9]. The conciseness and mathematical elegance of the formal semantic definition of a language is a very good measure of its conceptual integrity, and it may form a basis for proving the correctness of a certain implementation and function as a gauge for an equally formal theory of reasoning about the correctness of programs written in the language. Likewise, mobile systems also need formal method to formalize the description for mobile computation, till now many works are accomplished. Fox example, Cardelli gave the reduction rules for ambient in [1], Suna [10] gave the rewriting rules for CLAIM and so on. But all these formal descriptions are in operation semantics, though operation semantics description method has its own merits such as simple and intuitionist, it is indeterminate and difficult to implement automatization directly.

Here our concentration is denotational semantics [11]. Since the main character of denotational semantics is that it assigns a meaning to each language construct in a compositional way. Namely, the meaning of a composite construct only depends on the meanings of its constituents, and not on their actual syntactic form. Therefore, this is the best way of describing each concept in the language accurately and individually. It hasn't the flaws in operation semantics description method and it is easy to implement the automatical generation of mobile computation programs based on denotational semantics description. The main problem in describing mobile computation is that it is difficult to construct a uniform for all the components in mobile computation as before. Just because of mobility, it involves different behaviors and these behaviors may be executed in different computational contexts. So we can not construct the denotational semantics equations directly as traditional way. here by analyzing all kinds of behaviors in mobile computational system, we found three kinds of behaviors, they reside in an ambient, between ambients and in the whole system, thus we give a layered semantics description based the basic conception: ambient.

In section 2, we introduce the language and syntax of CLAIM. Following in section 3 we develop the layer denotational semantics for CLAIM, in this section we give the description for processes which respectively belonging to three levels: internal ambient, ambients and programs. At last we conclude our work in section 4.

## 2 CLAIM[10]

In this section, we present the specifications of CLAIM in a brief. An agent in CLAIM can be seen as a bounded place where the computation happens (similar to ambients) and has a list of local processes concurrently executed and a list of sub-agents. It has mental components such as knowledge, messages and goals, which enable a forward (reactive behavior) or a backward reasoning (goal driven behavior). In the programming language of CLAIM, agents and classes of agent are defined using:

```

defineAgent / defineAgentClass name {
    parent=null; | agentName ;
    knowledge=null; | {(knowledge;)*}
    goals=null; | {(goal;)*}
    messages=null; | { (queueMessage;)*}
    capabilities=null | { (capability;)*}
    processes=null | { (process |)*}
    agents=null | { (agentName;)*}
}

```

Where the parent component represents the name of the parent of the current agent, the agents are hierarchically organized, like the ambients in the ambient calculus; the knowledge component represents agent's knowledge about the other agents (i.e. about their capabilities and their class) or about the world (divers propositions); the goals component represents agent's current goals; the messages component represents agent's current messages queue, each agent has a messages queue for storing arrived messages. The messages are processed in their arrival order and are used to activate capabilities; the capabilities component represents the agents capabilities, the actions that he can do in order to achieve his goals or that he can offer to the other agents; the processes component represents agent's current processes executed in parallel. A process (P, Q, R...) can be a sequence of processes, a proposition, an instruction, a function (concerning the agent's components or defined in another programming language), the creation of a new agent, a mobility operation or a message transmission; the agent component represents the agent's sub-agents. For space, we only give short review of CLAIM and its reasoning and operational semantics can be seen in [10].

Our focus is on how a mobile computation system works, so the essential components about the mobility of CLAIM are our interesting point. As a result, we abstract the syntax of CLAIM as following and ignore other unnecessary ingredients:

$$\begin{aligned}
a \in \text{Amt} & ::= \text{Aid}[P, \text{AmtS}] \\
\text{Prog} & ::= \langle \text{Amt}_1 \parallel \dots \parallel \text{Amt}_n \rangle \\
p \in P & ::= 0 \mid \text{PIP} \mid P;P \mid X \cdot P \\
X & ::= \text{kill Amt} \mid \text{new Amt} \\
& \mid \text{out Amt} \mid \text{out-from Amt} \mid \text{out-to Amt} \\
& \mid \text{in Amt} \mid \overline{\text{in}} \text{ Amt} \\
& \mid \text{open Amt} \mid \overline{\text{open}} \text{ Amt}
\end{aligned}$$

And the reduction rules on CLAIM are defined as:

- (1)  $a [\text{kill } b_i \cdot p, \{b_1, b_2, \dots, b_i, \dots, b_n\}] \rightarrow a [p, \{b_1, b_2, \dots, b_n\}]$
- (2)  $a [\text{new } b \cdot p, A] \rightarrow a [p, A \cup \{b\}]$
- (3)  $a [\text{out-to}(c) p, \{\dots, b[\text{out-from}(c) \cdot q, \{\dots, c[\text{out } b \cdot r, A] \dots\}], \dots\}] \rightarrow a [p, \{\dots, b[q, \{\dots\}], c[r, A], \dots\}]$
- (4)  $a [\overline{\text{in}}(b) \cdot p, A] \parallel b [\text{in } a \cdot q, B] \rightarrow a [p, A \cup \{b[q, B]\}]$
- (5)  $a [\text{open } b \cdot p, \{\dots, b[\overline{\text{open}} a \cdot p, B], \dots\}] \rightarrow a [p \mid q, \{\dots, B\}]$

Where an ambient, semantically noted  $a[P, A]$ , has a name  $a$ , a set of running processes  $P$  and sub-agents  $A$ . Processes  $P$  are parallel of processes. A process can be empty or a sequence of capabilities, where a capability could be a creation or a kill of an agent, a mobility operation or an open action. “ $\parallel$ ” indicates the parallel of ambients and “ $\cdot$ ” indicates the parallel of processes. In order to specify secure distributed systems, mobility actions distinguish two levels of security according to the relative positions of the agents that want to communicate. Mobility inside a site is uncontrolled, as in the original ambient calculus: with actions in; out, an agent has the opportunity of entering an agent (in the same level in the hierarchy), or exiting his parent. Mobility through sites (move) is subject to control. Just as in the safe ambient calculus [12], we use co-actions to specify the control.

### 3 CLAIM’s Layered Denotational Semantics

Before our discussion, we first define some operations which are used in the paper.

**Define 3.1.** Operation  $\blacktriangleleft$ : Given that  $S = \{s_1, \dots, s_i, \dots, s_n\}$  is a finite set and  $s_i$  is an element of the set  $S$ , operation  $\blacktriangleleft$  applied on  $S$  and  $s_i$  is to delete  $s_i$  from  $S$ .

$$S \blacktriangleleft s_i = \{s_1, \dots, s_{i-1}, s_{i+1}, \dots, s_n\}$$

For example, if  $S = \{a, b, c\}$ , then  $S \blacktriangleleft a$  is  $\{b, c\}$

**Define 3.2.** Operation  $\blacktriangleright$ : Given that  $S = \{s_1, \dots, s_n\}$  is a finite set and an  $a$  which has the same type with the element of set  $S$  is not in set  $S$ , operation  $\blacktriangleright$  applied on  $a$  and  $S$  is to add  $a$  to  $S$ .

$$a \blacktriangleright S = \{a, s_1, \dots, s_n\}$$

For exmaple, if  $S = \{a, b\}$ ,  $c \blacktriangleright S$  is  $\{a, b, c\}$ .

### 3.1 Semantics for Processes in an Ambient

We introduce some sets first before defining the processes semantics equations. We define the set  $AState$  of internal state of an ambient, set  $Amt$  of active ambients names or null, set  $Cap$  and  $Co-Cap$  of capabilities and co-capabilities.

$$\begin{aligned} AState &= (\text{Var} \rightarrow \text{Val})^* \\ Cap &= \text{in} \cup \text{out} \cup \text{open} \\ Co-Cap &= \overline{\text{in}} \cup \text{out-to} \cup \text{out-from} \cup \overline{\text{open}} \end{aligned}$$

$AState$  is the internal state of a ambient that register the values of the variables of processes as in traditional programming. And we define the domain  $IProc$  of processes in an ambient.

$$ACont = AState \rightarrow IProc$$

$$\begin{aligned} IProc = \{p_0\} &\cup (AState \times IProc) \\ &\cup (Cap \times Amt \times AState \times ACont) \\ &\cup (Co-Cap \times Amt \times AState \times ACont) \end{aligned}$$

Where  $p_0$  is the terminated process;  $[\sigma, p]$  denotes the result of internal computation step,  $\sigma$  is the new state and  $p$  is the following process after this step;  $[cap, a, \sigma, f]$  indicates the mobility request to ambient  $a$ ,  $cap$  is a capability such as *in*, *out* or *open*, current state is  $\sigma$  and  $f$  is the resumption after this mobility step which is a function that receives an ambient state or resumption and returns a process;  $[co-cap, a, \sigma, f]$  describes mobile computation that co-cap may be any of *in*, *out-to*, *out-from* or *open*,  $a$  is an ambient,  $\sigma$  represents current state and the resumptions is  $f$ .

We define the semantics function and continuation function on internal processes as following,  $\mathbf{M}^E$  indicates the evaluation of expressions, just like the evaluation of conventional expressions. Here, we omit the definition of  $\mathbf{M}^E$ .  $\mathbf{M}^P$  is defined as:

$$\begin{aligned} \mathbf{M}^P : \quad & \text{Process} \rightarrow AState \rightarrow ACont \rightarrow IProc \\ & \mu \in ACont = AState \rightarrow IProc \\ \mathbf{M}^P[ \text{new } a[ \dots ] ] \sigma \mu &= [ \text{new}, a, \sigma, \lambda \sigma'. \mu(\sigma') ] \\ \mathbf{M}^P[ \text{kill } a ] \sigma \mu &= [ \text{kill}, a, \sigma, \lambda \sigma'. \mu(\sigma') ] \\ \mathbf{M}^P[ \text{in } a ] \sigma \mu &= [ \text{in}, a, \sigma, \lambda \sigma'. \mu(\sigma') ] \\ \\ \mathbf{M}^P[ \overline{\text{in}}(a) ] \sigma \mu &= [ \overline{\text{in}}, a, \sigma, \lambda \sigma'. \mu(\sigma') ] \end{aligned}$$

$$\begin{aligned}
\mathbf{M}^P[\text{out } a] \sigma \mu &= [\text{out}, a, \sigma, \lambda\sigma'. \mu(\sigma')] \\
\mathbf{M}^P[\text{out-to } (a)] \sigma \mu &= [\text{outto}, a, \sigma, \lambda\sigma'. \mu(\sigma')] \\
\mathbf{M}^P[\text{out-from } (a)] \sigma \mu &= [\text{outfrom}, a, \sigma, \lambda\sigma'. \mu(\sigma')] \\
\mathbf{M}^P[\text{open } a] \sigma \mu &= [\text{open}, a, \sigma, \lambda\sigma' \mu(\sigma')] \\
\mathbf{M}^P[\overline{\text{open } a}] \sigma \mu &= [\overline{\text{open}}, a, \sigma, p_0]
\end{aligned}$$

### 3.2 Semantics for Ambients

In last section we give the semantics description for internal processes in an ambient, in this section based on the discussion on internal level, we derive a higher level description by ignoring possible details that are no longer relevant at the ambient level.

We assume domain  $AProc$  of ambient processes. It is defined as:

$$\begin{aligned}
AProc = \{q_0\} \cup & (Amt \times Cap \times Amt \times AState \times (AState \rightarrow AProc)) \\
& \cup (Amt \times Co-Cap \times Amt \times AState \times (AState \rightarrow AProc))
\end{aligned}$$

We gain the semantics description of an ambient by an abstract function  $\partial$  that transforms internal description to ambient level. As a result, given an ambient, its semantics can be obtained by applying abstract function  $\partial$  to its processes part. The function  $\partial$  is defined as:

$$\begin{aligned}
\partial : IProc \times Amt &\rightarrow AProc \\
\partial(p_0) &= q_0 \\
\partial([\sigma, p]) &= \partial(p) \\
\partial([\text{cap}, b, \sigma, f]) \alpha &= [\alpha, \text{cap}, b, \sigma, \lambda\gamma. \partial(f(\gamma))] \\
\partial([\text{co-cap}, b, \sigma, f]) \alpha &= [\alpha, \text{co-cap}, b, \sigma, \lambda\gamma. \partial(f(\gamma))]
\end{aligned}$$

Therefore, given an ambient definition, we obtain semantics description on ambient level by:

$$\begin{aligned}
\mathbf{M}^A : Amt &\rightarrow AProc \\
\mathbf{M}^A[\text{amt}] &= \partial(\mathbf{M}^P[P] \sigma_0 \mu_0) a \\
\text{where } \text{amt} &= a[P, s_0] \\
\sigma_0 &= [\lambda x. \text{nil}, \{s_0\}] \\
\mu_0 &= \lambda\gamma. p_0
\end{aligned}$$

### 3.3 Semantics for Mobile Computing Programs

In above two sections, we don't describe how a new sub-ambient to be added or an old one to be killed, how the mobile actions to be computed. Hence in this section, we discuss this information to see how several ambients in parallel behave and



interact. Similar to above discussion, we first need to define the set  $PState$  of global state:

$$\begin{aligned} PState &= (Amt \rightarrow SubASet) \times (Amt \rightarrow CapQueue) \times (Amt \rightarrow FAmt) \\ CapQueue &= (Cap \times Amt \times AState \times (PState \rightarrow PProc)) \end{aligned}$$

In order to describe global behavior of mobile computing programs, we introduce another domain  $PProc$  of global processes.

$$PProc = \{r_0\} \cup (PState \times PProc)$$

Where  $r_0$  is the terminated process,  $[\rho, p]$  indicates that current state is  $\rho$  after last step and following step begins with  $p$ . namely the global behavior is embodied by the change of global state. We abstract global program description from ambient by introducing an operator  $\Phi: AProc \rightarrow PState \rightarrow PProc$ , which translates ambient processes into global processes when given an ambient and global state.  $\Phi$  is defined as:

$$\Phi : AProc \rightarrow PState \rightarrow PProc$$

$$\Phi (q_0) \rho = r_0$$

$$\begin{aligned} \Phi ([\alpha, \text{new}, b, \sigma, g])\rho &= \Phi (g(\sigma))\rho' \\ \rho'_{(1)}(\alpha) &= b \blacktriangleright \rho_{(1)}(\alpha) \\ \rho'_{(3)}(b) &= \alpha \\ \rho' &= [\rho'_{(1)}, \rho_{(2)}, \rho'_{(3)}] \end{aligned}$$

$$\begin{aligned} \Phi ([\alpha, \text{kill}, b, \sigma, g])\rho &= \Phi (g(\sigma))\rho' \\ \rho'_{(1)}(\alpha) &= \rho_{(1)}(\alpha) \blacktriangleleft b \\ \rho'_{(3)}(b) &= \text{None} \\ \rho' &= [\rho'_{(1)}, \rho_{(2)}, \rho'_{(3)}] \end{aligned}$$

$$\begin{aligned} \Phi ([\alpha, \text{in}, b, \sigma, g])\rho &= [\rho', r_0] \\ \rho'_{(2)}(\alpha) &= [\text{in}, b, \sigma, g] \blacktriangleright \rho_{(2)}(\alpha) \\ \rho' &= [\rho_{(1)}, \rho'_{(2)}, \rho_{(3)}] \end{aligned}$$

$$\Phi ([\alpha, \overline{\text{in}}, b, \sigma, g])\rho = \begin{cases} \Phi (g(\sigma))\rho' \mid \Phi (h(\sigma_b))\rho' & \text{if } (\rho_{(2)}(b)) = (\text{in}, \alpha, \sigma_b, h) \\ \text{wait} & \text{else} \end{cases}$$

$$\rho'_{(1)}(\alpha) = b \blacktriangleright \rho_{(1)}(\alpha)$$

$$\begin{aligned}
 \rho'_{(2)}(\alpha) &= \rho_{(2)}(\alpha) \blacktriangleleft [\text{in}, \alpha, \sigma_b, h] \\
 \text{fb} &= \rho_{(3)}(b) \\
 \rho'_{(1)}(\text{fb}) &= \rho_{(1)}(\text{fb}) \blacktriangleleft b \\
 \rho'_{(3)}(b) &= a \\
 \rho' &= [\rho'_{(1)}, \rho'_{(2)}, \rho'_{(3)}]
 \end{aligned}$$

$$\begin{aligned}
 \Phi([\alpha, \text{out}, b, \sigma, g]) \rho &= [\rho', r_0] \\
 \rho'_{(2)}(\alpha) &= [\text{out}, b, \sigma, g] \blacktriangleright \rho_{(2)}(\alpha) \\
 \rho' &= [\rho_{(1)}, \rho'_{(2)}, \rho_{(3)}]
 \end{aligned}$$

$$\begin{aligned}
 \Phi([\alpha, \text{outfrom}, b, \sigma, g]) \rho &= \begin{cases} [\rho', r_0] & \text{if } \rho_{(2)}(b) = [\text{out}, \alpha, \sigma_b, h] \\ \text{wait} & \text{else} \end{cases} \\
 \rho'_{(2)}(b) &= \rho_{(2)}(b) \blacktriangleleft [\text{out}, \alpha, \sigma_b, h] \\
 \rho''_{(2)}(b) &= [\text{outfrom}, \alpha, \sigma, k] \blacktriangleright \rho'_{(2)}(b) \\
 \rho' &= [\rho_{(1)}, \rho''_{(2)}, \rho_{(3)}] \\
 k &= \lambda \sigma'. g(\sigma) \mid h(\sigma_b)
 \end{aligned}$$

$$\begin{aligned}
 \Phi([\alpha, \text{outto}, b, \sigma, g]) \rho &= \begin{cases} \Phi(g(\sigma)) \rho' \mid \Phi(k(\sigma)) \rho' & \text{if } \rho_{(2)}(b) = [\text{outfrom}, a, \sigma_b, k] \\ \text{wait} & \text{else} \end{cases} \\
 \rho'_{(2)}(a) &= \rho_{(2)}(a) \blacktriangleleft b \\
 \rho'_{(3)}(b) &= \alpha \\
 \rho'_{(1)}(\alpha) &= b \blacktriangleright \rho_{(1)}(\alpha) \\
 \rho'_{(2)}(b) &= \rho_{(2)}(b) \blacktriangleleft [\text{outfrom}, a, \sigma_b, k] \\
 \rho' &= [\rho_{(1)}, \rho'_{(2)}, \rho'_{(3)}]
 \end{aligned}$$

$$\begin{aligned}
 \Phi([\alpha, \text{open}, b, \sigma, g]) \rho &= [\rho', r_0] \\
 \rho'_{(2)}(\alpha) &= [\text{open}, b, \sigma, g] \blacktriangleright \rho_{(2)}(\alpha) \\
 \rho' &= [\rho_{(1)}, \rho'_{(2)}, \rho_{(3)}]
 \end{aligned}$$

$$\begin{aligned}
 \Phi([\alpha, \overline{\text{open}}, b, \sigma, g]) \rho &= \begin{cases} \Phi(g(\sigma)) \rho' \mid \Phi(h(\sigma)) \rho' & \text{if } \rho_{(2)}(b) = [\text{open}, \alpha, \sigma_b, h] \\ \text{wait} & \text{else} \end{cases} \\
 \rho'_{(2)}(b) &= \rho_{(2)}(b) \blacktriangleleft [\text{open}, \alpha, \sigma_b, h]
 \end{aligned}$$

$$\begin{aligned}
\rho'_{(1)}(b) &= \rho_{(1)}(b) \blacktriangleleft \alpha \\
\rho''_{(1)}(b) &= \rho'_{(1)}(b) \cup \rho_{(1)}(\alpha) \\
\rho'_{(3)}(\rho_{(1)}(\alpha)) &= b \\
\rho'_{(3)}(\alpha) &= \text{None} \\
\rho' &= [\rho''_{(1)}, \rho'_{(2)}, \rho'_{(3)}]
\end{aligned}$$

Now, based on the discussion above, we give the semantics of mobile computation programs by the function  $\mathbf{M}^G$  which is defined as:

$$\mathbf{M}^G : \text{Prog} \rightarrow \text{PProc}$$

$$\mathbf{M}^G[\text{prog}] = \Phi(\partial(\mathbf{M}^P[p_1]\sigma_0\mu_0) a_1 \mid \partial(\mathbf{M}^P[p_2]\sigma_0\mu_0) a_2 \mid \dots \mid \partial(\mathbf{M}^P[p_n]\sigma_0\mu_0) a_n) \rho_0$$

Where  $\text{prog} = \langle A_1 \parallel A_2 \parallel \dots \parallel A_n \rangle$

$$\begin{aligned}
A_i &= a_i [p_i, s_0] \\
\sigma_0 &= [\lambda x. \text{nil}] \\
\mu_0 &= \lambda \gamma. p_0 \\
\rho_0 &= [A, \text{null}]
\end{aligned}$$

## 4 Proof

Now the question is how about our description framework, in order to prove the correctness of our semantics description framework, following we give the proof to test the sound of our approach. To achieve this, we apply this framework on the five reduction rules. For these reduction rules are the basis of mobile computation, each execution step is just depends on one of these reductions. So if we prove that it is correctness when this framework is applied on each rules

$$(1) a [ \text{new } b \cdot p, A ] \rightarrow a [ p, A \cup \{b\} ]$$

$$\text{That is to prove } \mathbf{M}^G [ a[\text{new } b \cdot p, A] ] \rho = \mathbf{M}^G [ a[p, A \cup \{b\}] ] \rho$$

While based on the equations defined above we have:

$$\begin{aligned}
\mathbf{M}^G [ a[\text{new } b \cdot p, A] ] \rho_0 &= \mathbf{M}^G [ a[\text{new } b \cdot p] ] \rho \\
&= \Phi(\partial(\mathbf{M}^P[\text{new } b \cdot p]\sigma_0\mu_0) a) \rho \\
&= \Phi(\partial([\text{new}, b, \sigma_0, \mu]) a) \rho \quad \text{where } \mu = \lambda \sigma. \mathbf{M}^P [ p ] \sigma \mu_0 \\
&= \Phi([a, \text{new}, b, \sigma_0, g]) \rho \quad \text{where } g = \lambda \gamma. \partial(\mu(\gamma)) \\
&= \Phi(g(\sigma_0)) \rho' \quad \text{where } \rho' = [\rho_{(1)}(\{b\} \cup A/a), \rho_{(2)}] \\
&= \Phi(\partial(\mathbf{M}^P [ p ] \sigma_0 \mu_0) a) \rho' \\
&= \mathbf{M}^G [ a[p, A] ] \rho'
\end{aligned}$$

$$(2) a [ \text{kill } b \cdot p, \{b\} \cup A ] \rightarrow a [ p, A ]$$

Just as (1), we have

$$\begin{aligned}
& \mathbf{M}^G [ a[\text{kill } b \cdot p] ] \rho = \Phi ( \partial ( \mathbf{M}^P [\text{kill } b \cdot p] \sigma_0 \mu_0 ) a ) \rho \\
& = \Phi ( \partial ( [\text{kill}, b, \sigma_0, \mu] ) a ) \rho \quad \text{where } \mu = \lambda\sigma. \mathbf{M}^P [ p ] \sigma \mu_0 \\
& = \Phi([a, \text{kill}, b, \sigma_0, g]) \rho \quad \text{where } g = \lambda\gamma. \partial (\mu(\gamma)) \\
& = \Phi(g(\sigma_0))\rho' \quad \text{where } \rho' = [\rho_{(1)}\{A/a\}, \rho_{(2)}] \\
& = \Phi ( \partial ( \mathbf{M}^P [ p ] \sigma_0 \mu_0 ) a ) \rho' \\
& = \mathbf{M}^G [ a[ p, A ] ] \rho
\end{aligned}$$

$$(3) a [ \overline{\text{in}} (b) \cdot p, A ] \parallel b [ \text{in } a \cdot q, B ] \rightarrow a [ p, A \cup \{b[q, B]\} ]$$

$$\begin{aligned}
& \mathbf{M}^G [ a[ \overline{\text{in}} (b) \cdot p, A ] \parallel b [ \text{in } a \cdot q, B ] ] \rho \quad \text{where } \mu_1 = \lambda\sigma. \mathbf{M}^P [ p ] \sigma \mu_0, \mu_2 = \lambda\sigma. \mathbf{M}^P [ q ] \sigma \mu_0 \\
& = \Phi([a, \overline{\text{in}}, b, \sigma_a, g_1]) \rho \mid \Phi([b, \text{in}, a, \sigma_b, g_2]) \rho \quad \text{where } g_1 = \lambda\gamma. \partial (\mu_1(\gamma)), g_2 = \lambda\gamma. \partial (\mu_2(\gamma)) \\
& = \Phi([a, \overline{\text{in}}, b, \sigma_a, g_1]) \rho' \mid [\rho', r_0] \quad \text{where } \rho' = [\rho_{(1)}, \rho_{(2)}\{\text{in}, a, \sigma_b, g_2\} / b\}] \\
& = \Phi(g_1(\sigma_a))\rho'' \mid \Phi(g_2(\sigma_b))\rho'' \quad \text{where } \rho'' = [\rho_{(1)}\{\{b\} \cup A/a\}, \rho_{(2)}] \\
& = \Phi ( \partial ( \mathbf{M}^P [ p ] \sigma_a \mu_0 ) a ) \rho'' \mid \Phi ( \partial ( \mathbf{M}^P [ q ] \sigma_b \mu_0 ) b ) \rho''
\end{aligned}$$

$$(4) a [ \text{out-to}(c) \cdot p, \{b[\text{out-from}(c) \cdot q, \{c[\text{out } b \cdot r, A]\}]\} ] \rightarrow a[p, \{b[q], c[r, A]\}]$$

$$\begin{aligned}
& \mathbf{M}^G [ a [ \text{out-to}(c) \cdot p, \{b[\text{out-from}(c) \cdot q, \{c[\text{out } b \cdot r, A]\}]\} ] ] \rho \\
& = \Phi ( \partial ( \mathbf{M}^P [\text{out-to}(c) \cdot p] \sigma_a \mu_0 ) a \mid \partial ( \mathbf{M}^P [q] \sigma_b \mu_0 ) b \mid \partial ( \mathbf{M}^P [r] \sigma_c \mu_0 ) c ) \rho \\
& = \Phi([a, \text{outto}, c, \sigma_a, g_a]) \rho \mid \Phi([b, \text{outfrom}, c, \sigma_b, g_b]) \rho \mid \Phi([c, \text{out}, b, \sigma_c, g_c]) \rho \\
& \quad \text{where } \mu_a = \lambda\sigma. \mathbf{M}^P [ p ] \sigma \mu_0, \mu_b = \lambda\sigma. \mathbf{M}^P [ q ] \sigma \mu_0, \mu_c = \lambda\sigma. \mathbf{M}^P [ r ] \sigma \mu_0 \\
& \quad g_a = \lambda\gamma. \partial (\mu_a(\gamma)), g_b = \lambda\gamma. \partial (\mu_b(\gamma)), g_c = \lambda\gamma. \partial (\mu_c(\gamma)) \\
& = \Phi([a, \text{outto}, c, \sigma_a, g_a]) \rho' \mid \Phi([b, \text{outfrom}, c, \sigma_b, g_b]) \rho' \mid [\rho', r_0] \\
& \quad \text{where } \rho' = [\rho_{(1)}, \rho_{(2)}\{\{\text{out}, b, \sigma_c, g_c\}/c\}] \\
& = \Phi([a, \text{outto}, c, \sigma_a, g_a]) \rho'' \mid [\rho'', r_0] \quad \text{where } \rho'' = [\rho'_{(1)}, \rho'_{(2)}\{\{\text{outfrom}, b, \sigma_b, k\}/c\}] \\
& \quad k = \lambda\gamma. (g_b(\sigma_b) \mid g_c(\sigma_c))
\end{aligned}$$

$$= \Phi(g_a(\sigma_a))\rho'' \mid \Phi(k(\sigma_a))\rho''$$

$$= \Phi ( \partial ( \mathbf{M}^P [ p ] \sigma_a \mu_0 ) a ) \rho'' \mid \Phi ( \partial ( \mathbf{M}^P [ q ] \sigma_b \mu_0 ) b ) \rho'' \mid \Phi ( \partial ( \mathbf{M}^P [ r ] \sigma_c \mu_0 ) c ) \rho''$$

$$(5) a [ \text{open } b \cdot p, \{b[ \overline{\text{open } a \cdot q} ]\} ] \rightarrow a [ p \mid q ]$$

$$\mathbf{M}^G [ a [ \text{open } b \cdot p, \{b[ \overline{\text{open } a \cdot q} ]\} ] ] \rho$$

$$= \Phi ( \partial ( \mathbf{M}^P [ \text{open } b \cdot p ] \sigma_a \mu_0 ) a ) \rho \mid \Phi ( \partial ( \mathbf{M}^P [ \overline{\text{open } a \cdot q} ] \sigma_b \mu_0 ) b ) \rho$$

$$= \Phi([a, \text{open}, b, \sigma_a, g_a]) \rho \mid \Phi([b, \overline{\text{open}}, a, \sigma_b, g_b]) \rho$$

$$\text{where } \mu_a = \lambda\sigma. \mathbf{M}^P [ p ] \sigma \mu_0, \mu_b = \lambda\sigma. \mathbf{M}^P [ q ] \sigma \mu_0$$

$$\begin{aligned}
& g_a = \lambda\gamma. \partial (\mu_a(\gamma)), g_b = \lambda\gamma. \partial (\mu_b(\gamma)) \\
& = [\rho', r_0] \mid \Phi(\overline{[b, \text{open}, a, \sigma_b, g_b]}) \rho' \quad \text{where } \rho' = [\rho_{(1)}, \rho_{(2)}\{\{\text{open}, b, \sigma_a, g_a\}/a\}] \\
& = \Phi(g_b(\sigma_b))\rho'' \mid \Phi(g_a(\sigma_a))\rho'' \quad \text{where } \rho''_{(1)}(a) = \rho'_{(1)}(a) \blacktriangleleft b, \rho'' = [\rho_{(1)}'', \rho_{(2)}] \\
& = \Phi(\partial(\mathbf{M}^P[p] \sigma_a \mu_0) a) \rho'' \mid \Phi(\partial(\mathbf{M}^P[q] \sigma_a \mu_0) a) \rho'' \\
& = \mathbf{M}^G[a[p|q]] \rho''
\end{aligned}$$

## 5 Conclusion

In the paper we talk about how to describe mobile computation in denotational semantics. It is known that description in operational semantics is easy to obtain, however, it brings the indetermination and is difficult to implement. As a result it is necessary indeed to give a denotational semantics description for mobile computation. By analyzing the variant behaviors in mobile computation, we give a layered denotational semantics which is defined at three different levels: for internal process of an ambient, of ambients and of programs. For each of these levels we define the semantics domains and semantics description equations as well as the transformation relationship between levels. From the paper, an explicit model of the behaviors in mobile computation can be obtained. Here we take CLAIM as an example, since CLAIM is a computational language for autonomous, intelligent and mobile agents, and it embodies the essential characters in mobile computation.

The next step of our work will try to give denotational semantics description for our practical application system in this model. Based on the semantics model in the paper, we hope that we could give a formal description for our system, which is important to optimize and improve the efficiency of our system.

## References

1. Luca Cardelli, Andrew D. Gordon. Mobile Ambient. In: Proceedings of FOSSACS'98. Lecture notes in computer science, Vol. 1378. Springer, Berlin Heidelberg New York, pp 140–155.
2. Luca Cardelli, Andrew D. Gordon. Types for Mobile Ambients. In Proc. 26th POPL, pages 79-92. ACM Press, 1999.
3. Yoav Shoham. Agent Oriented Programming. Artificial Intelligence (60), pages 51-92, 1993.
4. Rebecca S.Thomas. The PLACA Agent Programming Language. Proceedings of the workshop on agent theories, architectures, and languages on Intelligent agents, pages 355-370, 1995.
5. Gerd Wagner. VIVA Knowledge-Based Agent Programming. Preprint (on-line at: [www.inf.fu-berlin.de/wagner/VIVA.ps.gz](http://www.inf.fu-berlin.de/wagner/VIVA.ps.gz)), 1996.
6. K.V.Hindriks, F.S.deBoer, W.van der Hoek, J.J.Ch.Meyer. Agent Programming in 3APL. Intelligent Agents and Multi-Agent Systems, Vol. 2, pages 357-401, 1999.

7. A. El Fallah-Seghrouchni, A. Suna. An Unified Framework for Programming Autonomous, Intelligent and Mobile Agents. LNAI, Vol. 2691, 2003, pages 353-362.
8. A. El Fallah-Seghrouchni, A. Suna. CLAIM: A Computational Language for Autonomous, Intelligent and Mobile Agents, LNAI, Vol. 3067, 2004.
9. Pierre America. The Practical Importance of Formal Semantics. In J.W. Klop, J.-J. Ch. Meyer, and J. J. M. M. Rutten, editors, *J. W. de Bakker, 25 Jaar Semantiek, Liber Amicorum*, pages 31-40. Centre for Mathematics and Computer Science, Amsterdam, the Netherlands, April 1989.
10. Alexandru Suna, A. El Fallah-Seghrouchni, Christophe Fouquer, Patrick Baillot. Mobile Multi-agent Systems: A Programming Language and Its Semantics. Third International Joint Conference on Autonomous Agents and Multi-agent Systems (AAMAS'04), New York, USA, Volume 3. 2004, pages 1386-1387.
11. Michael J.C. Gordon. *The Denotational Description of Programming Languages: An Introduction*. Springer, 1979.
12. F. Levi and D. Sangiorgi. Controlling Interference in Ambients. In Proceedings of the 27th ACM SIGPLAN-SIGACT Symposium, 2000, pages 352-364.

# Immunity and Mobile Agent Based Intrusion Detection for Grid

Xun Gong, Tao Li, Ji Lu, Tiefang Wang, Gang Liang, Jin Yang, and Feixian Sun

School of Computer Science, Sichuan Univ., Chengdu 610065, China  
orchid\_xun@tom.com

**Abstract.** This paper analyzes the distinctive characteristics of grid environments and proposes a novel immunity and mobile agent based intrusion detection for grid (*IMIDG*) model. Then, the concepts and formal definitions of *self*, *nonsel*, *antibody*, *antigen*, *agent* and *match algorithm* in the grid security domain are given. Besides, the mathematical models of *self*, *mature MoA* (mature monitoring agent), *dynamic memory MoA* (memory monitoring agent) *survival*, *CoA* (communicator agent), and *BoA* (beating off agent) are established. The effects of several import parameters on system performance and detection efficiency in the model of dynamic memory MoA survival are analyzed and shown in the experiments. Our theoretical analysis and experimental results show the model which enhances detection efficiency and assures steady performance in immune-based IDS is a good solution to grid intrusion detection.

## 1 Introduction

Grid Computing shows a nice future, but also presents tough security challenges. A grid environment has many distinctive characteristics that are different from those of common network environments [1], so the mechanisms and policies for securing the grid are more complicated.

Intrusion detection is an important component of network security systems. Intrusion detection systems are designed to detect unauthorized use, misuse and abuse of computer system from internal and external intruders. So intrusion detection also is needed to provide an additional defense layer in the grid security systems. There has been abundant research on intrusion detection for common network environments, but current research on grid intrusion detection is still in its infancy, and a few references can be used [2] [3]. The characteristics of grid lead to intrusion detection problems that are not addressed by existing intrusion detection technologies for common network environments.

Artificial immune system has the features of dynamic, self-adaptation and diversity [4-10] that just meet the constraints derived from the characteristics of the grid environment, and mobile agent has many same appealing properties as that of artificial immune system [11][12]. Thus, we apply mobile agent technology as support for intrusion detection and propose a novel immunity and mobile agent based intrusion detection for grid (*IMIDG*) model.

In *IMIDG*, the concepts and formal definitions of *self*, *nonsel*, *antibody*, *antigen* and *agent* (that simulates the lymphocyte and is used as a detector to recognize non-self antigens, i.e. intrusions,) in the grid security domain are given firstly. We define

three kinds of agents: monitoring agents (*MoA*), communicator agents (*CoA*) and beating off agents (*BoA*). *MoAs* simulate B-lymphocytes in the immune system and patrol grid nodes. They are responsible for monitoring various parameters simultaneously at four levels (user level, system level, process level, and packet level) and detecting intrusion. As B-lymphocytes consist of mature and memory lymphocytes, *MoAs* are divided into mature and memory *MoAs* that simulate, respectively, mature and memory cells. *CoAs* serve as communicators and are responsible for message transmission among agents. They simulate lymphokines secreted from T cells to stimulate B cells to clone themselves or eliminate cloned cells. *BoAs* simulate T killer cells are responsible for dealing with intrusive activities. Once *MoAs* detect intrusions, they will stimulate *CoAs* and present the features of intrusions to *BoAs*. *CoAs* will activate *BoAs*. *BoAs* will move to the suspected place to counterattack intrusions. The counterattack strategies consist of disconnecting a node, killing a process, discarding dubitable packets, and so on. Then, the mathematical models of *self*, *mature MoA*, *dynamic memory MoA survival*, *CoA* and *BoA* are established.

In the present immunity based intrusion detection systems [13], the memory lymphocytes have an unlimited lifecycle except they match the newly added selfs. Obviously, a considerable number of memory lymphocytes will be generated in the end. According to the above analysis, the total number of cells is the sum of the mature and memory cells numbers. Mature cells either evolve into memory ones or die before they exceed the lifecycle, so the number of mature cells remains steady and does not exceed a certain value. The size of memory cells definitively affects the total number of cells. As the increasing of memory cells, the cells number will increase, which will increase the intrusion detection time. When the number of cell reaches a certain value, the intrusion detection system will either become a bottleneck or ineffective as some packets are skipped.

However, *IMIDG* introduces a new idea of *dynamic memory MoAs survival* to overcome this problem: a given number of the least recently used memory *MoAs* that simulate memory lymphocytes will degrade into mature *MoAs* and be given a new age and count, if the number of *MoAs* reaches or exceeds a given maximum. In mature *MoAs*, once a degraded memory *MoA* matches only one antigen in its lifecycle, it will be activated immediately and evolve into memory one again. The method assures the memory cell number does not exceed a given value, so that the total number of cells does not exceed a given value. So it does not take too long time to detect intrusions. It is important to set the number of degraded memory *MoAs* to a given value, which assures *IMIDG* steady performance. If the number of degraded memory *MoAs* is not given, random number of degraded memory *MoAs* will degrade into mature ones and left small or large remainder. The false negative rate will be large or intrusion detection time will be long. And degraded memory *MoAs* have priority in evolvment, which enhances detection efficiency.

## 2 The Proposed Intrusion Detection for Grid Model (*IMIDG*)

*IMIDG* has two sub-models (see Fig. 1). The first is the model of the generation and evolvment of *MoAs* in the square frame. The second is the intrusion detection model.



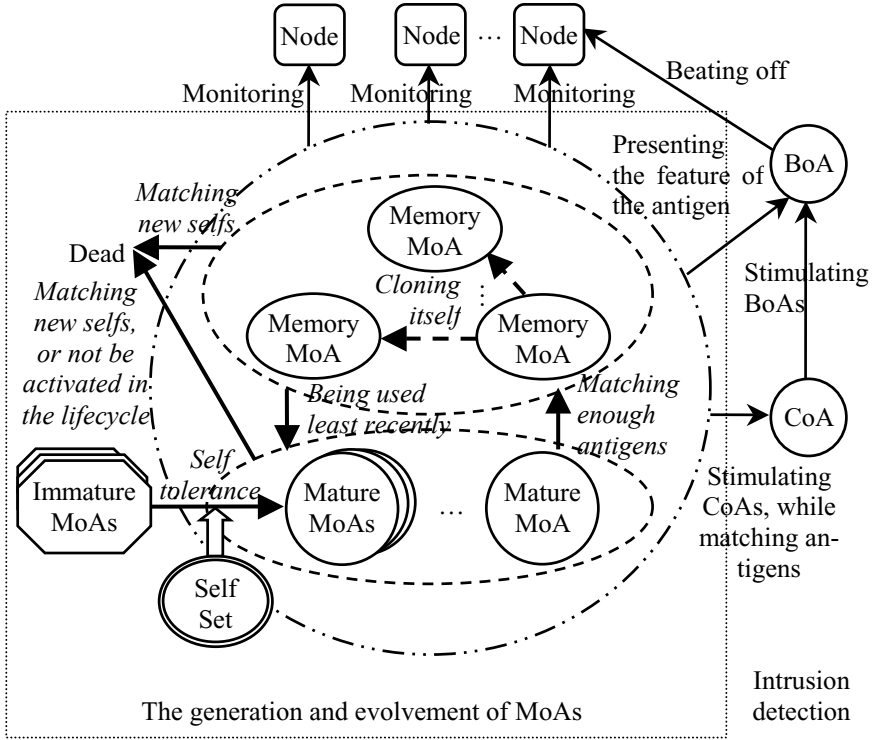


Fig. 1. The proposed intrusion detection for grid model

In the first model, we define antigens ( $Ag$ ) to be the features of grid services and accesses, and given by:

$$Ag = \{ag \mid ag \in D\}. \tag{1}$$

$$D = \{d \mid d = \langle user\_level, system\_level, process\_level, packet\_level \rangle\}. \tag{2}$$

$$D = \{0,1\}^{12l}. \tag{3}$$

$$user\_level = \{0,1\}^{3l}. \tag{4}$$

$$system\_level = \{0,1\}^{3l}. \tag{5}$$

$$process\_level = \{0,1\}^{3l}. \tag{6}$$

$$packet\_level = \{0,1\}^{3l}. \tag{7}$$

*user\_level*, *system\_level*, *process\_level* and *packet\_level* present the parameters, which are monitored by monitoring agents at user, system, process, and packet level respectively. They are also called, respectively, the field of an antigen. *l* is a natural number (constant).

The user level parameters monitored to detect intrusion include:

- a. Type of user and user privileges
- b. Access of resources
- c. Type of software/programs use

The system level parameters monitored to detect intrusion include:

- d. Cumulative and per user CPU usage
- e. Usage of real and virtual memory
- f. I/O and disk usage

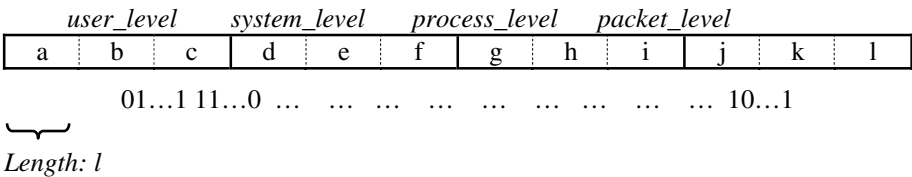
The process level parameters monitored to detect intrusion include:

- g. The number of processes and their types
- h. Current state of the process (running, blocked, waiting) and runaway processes
- i. Percentage of various process times (such as user process time, system process time and idle time)

The process level parameters monitored to detect intrusion include:

- j. Number, duration, type (Remote/Local) and status of connections (e.g. established, close\_wait, time\_wait)
- k. Average number of packets sent and received
- l. Protocol and port used [11]

The twelve parameters are grouped into a set and expressed in binary codes of the same length *l*. The architecture of *D* is given in the following.



**Fig. 2.** The architecture of *D*

Obviously, the lengths of field *user\_level*, *system\_level*, *process\_level* and *packet\_level* are all *3l*. The length of string *ag* is *12l*.

For the convenience using the fields of an antigen *x*, a subscript operator “.” is used to extract a specified field of *x*, where

$$x.fieldname = \text{the value of field } fieldname \text{ of } x. \tag{8}$$

For example,  $x.user\_level = 01101...10$ .

We define *Self* to be the set of normal grid services and accesses. Similarly, *Non-self* is a set of abnormal services and accesses. *Ag* contains two subsets, *Self* and *Non-self*, where  $Self \subset Ag$  and  $Nonself \subset Ag$  such that

$$Self \cup Nonself = Ag, \quad Self \cap Nonself = \Phi. \quad (9)$$

In the intrusion detection model, all the agents form a set (*Agent*). *Agent* contains three elements, *MoA*, *CoA* and *BoA*. Monitoring agents, communicator agents and beating off agents form, respectively, the set *MoA*, *CoA* and *BoA*. A monitoring agent is used as a detector to recognize nonself antigens (intrusions). Thus, we have:

$$Agent = \{MoA, CoA, BoA\}. \quad (10)$$

$$MoA = \{ \langle d, age, count \rangle \mid d \in D, age \in N, count \in N \}. \quad (11)$$

*d* is the lymphocyte antibody that is used to match an antigen. Each monitoring agent carries several antibodies. *age* is the agent age, *count* (affinity) is the antigen number matched by antibody *d*, and *N* is the set of natural numbers. *d*, *age* and *count* are also called, respectively, field *d*, *age* and *count* of an agent. Similarly, the subscript operator “.” is used to extract a specified field of an agent *x*. For example,  $x.d.user\_level = 1101\dots101$ ,  $x.age = 25$ .

$$CoA = \{c\} \quad (12)$$

If a MoA stimulates a CoA, *c* will be set to 1.

$$BoA = \{ \langle b, action \rangle \mid b \in D \} \quad (13)$$

*d*, *action* are, respectively, the fields of the BoA. If a CoA stimulates a BoA, *b* will be set to 1.

MoAs consist of Mature and Memory ones (see sect. 1). A mature MoA is a MoA that is tolerant to self but is not activated by antigens. A memory MoA evolves from a mature one that matches enough antigens in its lifecycle. Mature and memory MoAs form, respectively, the set  $MA_{MoA}$  and  $ME_{MoA}$ . Mature MoAs detect novel intrusions that have not previously been identified. Memory MoAs greatly enhance detection of previously seen intrusions. Therefore, we have:

$$MoA = MA_{MoA} \cup ME_{MoA}. \quad (14)$$

$$MA_{MoA} \cap ME_{MoA} = \Phi. \quad (15)$$

$$MA_{MoA} = \{ x \mid x \in MoA, \forall y \in Self \ ( \langle x.d, y \rangle \notin Match \wedge x.count < \beta ) \}. \quad (16)$$

$$ME_{MoA} = \{ x \mid x \in MoA, \forall y \in Self \ ( \langle x.d, y \rangle \notin Match \wedge x.count \geq \beta ) \}. \quad (17)$$

*Match* is a match relation in *D* defined by

$$\begin{aligned} Match = \{ \langle x, y \rangle \mid (x, y \in D), & (f_{match}(x.user\_level, y.user\_level) = 1 \\ \vee f_{match}(x.system\_level, y.system\_level) = 1 & \\ \vee f_{match}(x.process\_level, y.process\_level) = 1 & \\ \vee f_{match}(x.process\_level, y.p) = 1) \}. & \end{aligned} \quad (18)$$

$f_{match}(x, y)$  is based on the affinity between  $x$  and  $y$ : if the affinity greater than a specified threshold, then 1 is returned, otherwise, 0 is returned. In *IMIDG*, the affinity function can be *r-contiguous-bits matching rule*, *Hamming distance*, *Landscape-Affinity Matching*, etc. As intrusions do not always happen simultaneously at four levels, we consider an intrusion happens as long as MoA matches an antigen at any one level of the fours. The method enhances detection efficiency.

MoAs can clone themselves if the intrusion is detected. Some cloned MoAs are left here to detect more intrusions, and others are moved to other grid nodes to detect the similar intrusions.

## 2.1 The Definition of Self

In a dynamic grid environment, as time goes on, some grid services and accesses, which were normal in the past, are forbidden now. Equation (19) depicts the evolution of self, where  $Self(t)$  evolves from  $Self(t-1)$ , the mutated self antigens ( $Self_{variation}$ ) at time  $t$  are eliminated, and the newly defined self antigens ( $Self_{new}$ ) at time  $t$  are added.

$$Self(t) = \begin{cases} \{x \mid x \text{ is the initial self antigen} \}, & t = 0 \\ Self(t-1) - Self_{variation}(t) \cup Self_{new}(t), & t \geq 1 \end{cases} \quad (19)$$

$$Self_{variation}(t) = \{x \mid x \text{ is the self antigen forbidden at time } t\}. \quad (20)$$

$$Self_{new}(t) = \{y \mid y \text{ is the self antigen added at time } t\}. \quad (21)$$

## 2.2 Mature MoAs Model

$$MA_{MoA}(t) = \begin{cases} \Phi, & t = 0 \\ MA_{retain}(t) \cup MA_{new}(t) \cup MA_{cycle}(t) - MA_{activation}(t), & t \geq 1 \end{cases} \quad (22)$$

$$MA_{retain}(t) = MA'_{MoA}(t) - A(t) \cup A'(t). \quad (23)$$

$$MA'_{MoA}(t) = \{y \mid y \in MoA, x \in MA_{MoA}(t-1), x.age < \lambda, \forall s \in Self_{new}(t) < x.d, s > \notin Match, y.d = x.d, y.age = x.age + 1, y.count = x.count\}. \quad (24)$$

$$A(t) = \{x \mid x \in MA'_{MoA}(t), \exists y \in Ag(t-1) < x.d, y > \in Match\}. \quad (25)$$

$$A'(t) = \{y \mid y \in MoA, x \in A(t), c \in CoA(t), w \in BoA(t), y.d = x.d, y.age = x.age, y.count = x.count + 1, c = 1, w.b = x.d\}. \quad (26)$$

$$MA_{cycle}(t) = \{y \mid y \in MoA, y.d = x.d, y.age = T, y.count = \beta - 1, x \in ME_{degradation}(t), T < \lambda\}. \quad (27)$$

$$MA_{activation}(t) = \{x \mid x \in A'(t), x.count \geq \beta\}. \quad (28)$$

$$MA_{new}(t) = \{y \mid y \in MoA, y.d = x.d, y.count = 0, y.age = 0, \forall s \in Self(t) < x, s > \notin Match\}. \quad (29)$$

$$I_{new}(t) = \begin{cases} \Phi, & |MA_{MoA}(t-1)| \geq \Gamma \\ \{y_1, y_2, \dots, y_{\Gamma - |MA_{MoA}(t-1)|}\}, y_i \in D, 1 \leq i \leq \Gamma - |MA_{MoA}(t-1)|, & otherwise \end{cases} \quad (30)$$

$MA_{retain}(t)$  simulates the process that the mature cells evolve into the next generation ones, where the cells do not tolerate to those newly added self elements or have not match enough antigens ( $\beta > 0$ ) in lifecycle  $\lambda$ , will be eliminated. If a MoA match an antigen, it will stimulate a CoA and set  $c$  of the CoA to 1. It will also present the feature of the antigen (field  $d$  of the MoA) to a BoA. So  $w.b$  is set to  $x.d$ .  $MA_{new}(t)$  depicts the generation of new mature MoAs.  $\Gamma (>0)$  is the max number of mature MoAs in *IMIDG*.  $MA_{activation}(t)$  is the set of mature MoAs which match enough antigens and will be activated and evolve into memory MoAs at time  $t$ .  $MA_{cycle}(t)$  is the set of the least recently used memory MoAs which degrade into mature MoAs and be given a new age  $T (>0)$  and count  $\beta-1$  ( $\beta \geq 1$ ). Once a mature MoA with count  $\beta-1$  matches an antigen in lifecycle  $\lambda$ , the count of it will be set to  $\beta$  and it will be activated again. Because the degraded memory MoA has better detection capability than mature MoAs, it is endowed with a priority in evolvement into memory MoAs. When the same antigens arrive again, they will be detected immediately by the re-evolved memory MoA. The method enhances detection efficiency. If the value of  $T$  is too large and even close to  $\lambda$ , the degraded memory MoA will die soon because it cannot match enough antigens in lifecycle  $\lambda$ . But, if the value of  $T$  is too small and even close to 0, the degraded memory MoA will have been in mature MoAs for a long time, and the number of mature MoAs will reach the maximum  $\Gamma$ . Here, immature MoAs will not be generated, which will affect the diversity of *IMIDG*.

### 2.3 Dynamic Memory MoAs Survival Model

$$ME_{MoA}(t) = \begin{cases} \Phi, & t = 0 \\ ME_{retain}(t) \cup ME_{new}(t) - ME_{degradation}(t), & t \geq 1 \end{cases} \quad (31)$$

$$ME_{retain}(t) = ME'_{MoA}(t) - E(t) \cup E'(t). \quad (32)$$

$$ME'_{MoA}(t) = \{y \mid y \in MoA, x \in ME_{MoA}(t-1), \forall s \in Self_{new}(t) < x.d.s > \notin Match, y.d = x.d, y.age = x.age + 1, y.count = x.count\}. \quad (33)$$

$$E(t) = \{x \mid x \in ME'_{MoA}(t), \exists y \in Ag(t-1) < x.d, y > \in Match\}. \quad (34)$$

$$E'(t) = \{y \mid y \in MoA, x \in E(t), c \in CoA(t), w \in BoA(t), y.d = x.d, y.age = 0, y.count = x.count + 1, c = 1, w.b = x.d\}. \quad (35)$$

$$ME_{new}(t) = \{x \mid x \in ME_{MoA}, y \in MA_{activation}(t), x.d = y.d, x.age = 0, x.count = y.count\}. \tag{36}$$

$$ME_{degradation}(t) = \begin{cases} \{x_1, x_2, \dots, x_p\}, 1 \leq i \leq P, x_i \in ME_{degradation}(t), \\ \forall y \in ME_{retain}(t) - ME_{degradation}(t), y.age \leq x_i.age, |ME_{MoA}(t-1)| \geq \kappa \\ \Phi, \end{cases} \tag{37}$$

otherwise

Equation (31) depicts the dynamic evolvement of memory MoAs, where the memory MoAs that match the newly added selfs will die and those matched by an antigen will be activated immediately. If a memory MoA matches an antigen, it will stimulate a CoA and present the feature of the antigen (the field *d* of the MoA) to a BoA.  $ME_{retain}(t)$  simulates the process that the memory MoAs evolve into the next generation ones.  $ME_{new}(t)$  is the set of new memory MoAs evolved from mature ones.  $ME_{degradation}(t)$  is the set of memory MoAs that are not activated by antigens lately and are selected randomly to degrade into mature MoAs when the number of memory MoAs reaches or exceeds a given maximum  $\kappa$ .  $P (>0)$  is the number of selected memory MoAs. Obviously, the number of memory MoAs does not exceed a given value.

### 2.4 CoA Model

$$CoA(t) = \begin{cases} \{c \mid c = 0\}, & t = 0 \\ \{c \mid w \in BoA(t-1), \exists c \in CoA(t-1) c = 1, w.action = 1, c = 0\}, & t \geq 1 \end{cases} \tag{38}$$

Equation (38) depicts the process that the CoA transmits message among agents. When  $t=0$ , *c* is set to 0. If a CoA stimulates a BoA, it will set the field *action* of the BoA to 1 and reset the field *c* of itself to 0.

### 2.5 BoA Model

$$BoA(t) = \begin{cases} \{w \mid wb = \langle 0 \dots 0, 0 \dots 0, 0 \dots 0, 0 \dots 0 \rangle, w.action = 0\}, & t = 0 \\ \{w \mid \exists w \in BoA(t-1) (wb \neq \langle 0 \dots 0, 0 \dots 0, 0 \dots 0, 0 \dots 0 \rangle \wedge w.action = 1), \\ \quad wb = \langle 0 \dots 0, 0 \dots 0, 0 \dots 0, 0 \dots 0 \rangle, w.action = 0\}, & t \geq 1 \end{cases} \tag{39}$$

If a CoA stimulates a BoA, the BoA will beat off the intrusion according to the feature of antigen that the MoA presents. When the BoA finishes its work, it will reset the field *b* of itself to  $\langle 0 \dots 0, 0 \dots 0, 0 \dots 0, 0 \dots 0 \rangle$  and *action* to 0.

## 3 Simulations and Experiment Results

To prove the intrusion detection performance (mainly including error rate and intrusion detection time) of *IMIDG*, we developed abundant experiments. The experiments were carried out in the Laboratory of Computer Network and Information Security at

Sichuan University. We developed a grid simulation toolkit based on GridSim [14] to satisfy our needs. The simulation environment simulates users with different behaviors, resources and agents. A total of 40 computers in a grid were under surveillance. The length of selfs was 1 K. The affinity threshold was 0.7. As the size of memory cells definitively affects the total number of cells that affects the intrusion detection time, we only used memory MoAs to detect intrusions in order to simplify the experiments. Every memory MoA carries an antibody. The initial number of memory MoAs was 32. 32 memory MoAs were generated at a time. The memory MoAs maximum  $\mathcal{K}$  was set to 512. The numbers of memory MoAs at the seventh and the fifteenth time point reach 256 and 512, respectively.

To prove the importance of parameter P in the dynamic memory MoAs survival model, we developed experiments to show the effect of P on error rate (false positive rate and false negative rate) in *IMIDG*. In the first set of experimentations, P was set to 288, 352, 416, 480, respectively. The results are shown in Fig.3. The false positive (FP) rate nearly retained 0 in spite of the numbers of memory MoAs and P at all times. The false negative (FN) rate was 0.38 when the number of memory MoAs was 32 in the beginning. The false negative rate dropped to 0, when the number of memory MoAs increased to 256 at the seventh time point. When P = 288, 288 memory MoAs degraded into mature ones when the number of memory MoAs reached the given maximum 512 at the fifteenth time point and at the twenty-fourth time point again but the false negative rate retained 0. There was not a fluctuation in the false negative rate curve. When 352, 416, or 480 memory MoAs degraded into mature ones at the fifteenth and twenty-fourth time point, the false negative rate, respectively, increased to 0.05, 0.17 and 0.28 at the sixteenth and twenty-fifth time point. Because memory MoAs numbers, respectively, were 192, 128 and 64 at the sixteenth and twenty-fifth time point. The fluctuation in the three false negative rate curves was obvious. The four curves of false negative rate show that it assures the lowest false negative rate in *IMIDG* to set P to 288 when  $\mathcal{K}$  was 512. So the experiments results prove it assures the steady performance easily to select an appropriate number of degraded memory MoAs according to the given maximum of memory MoAs.

To prove that the dynamic memory MoAs survival model reduces the intrusion detection time of *IMIDG*, we developed experiments to compare the intrusion detection time of *IMIDG* with that of current immunity-based intrusion detection systems (IDS) [13]. In the second set of experimentations, P was set to 288 because the lowest false positive rate and false negative rate were assured according to the first set of experimentations. The results are shown in Fig.4. 288 least recently used memory MoAs degraded into mature MoAs when the number of memory MoAs reached the given maximum 512 at the fifteenth time point and at the twenty-fourth time point again in *IMIDG*. The detecting time was reduced rapidly at the sixteenth and twenty-fifth time point in *IMIDG*, while grew almost linearly in current IDSs. Obviously, the number of memory MoAs that fluctuated between 512 and 224 did not exceed the given maximum 512. The detecting time increased to and did not exceed the maximum 16.5 when the number of memory MoAs reached the given maximum 512. So the set of experiment results illuminate the dynamic memory MoAs survival model reduces the intrusion detection time of *IMIDG*, while the intrusion detection time in current IDSs grows almost linearly due to the increasing of memory cells.

Therefore, the two set of experiments results show that the dynamic memory MoAs survival model assures steady detection performance and high detection efficiency, and keeps balance between error rate and intrusion detection time in *IMIDG*.

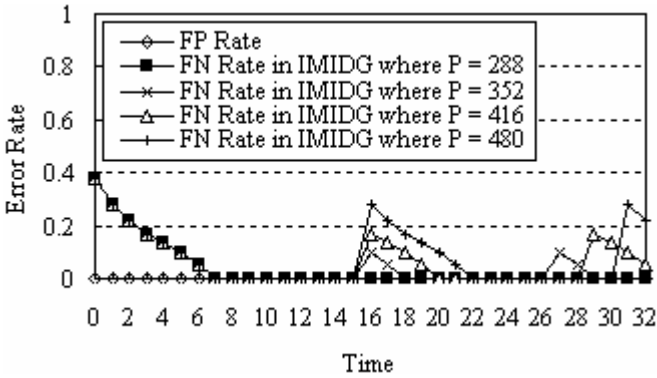


Fig. 3. Effect of P on error rate in *IMIDG*

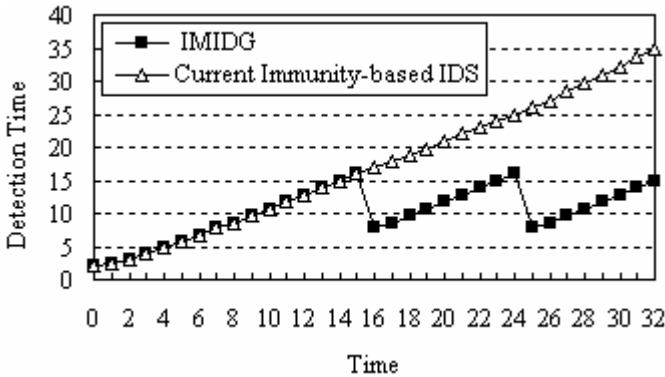


Fig. 4. Effect of the number of memory MoAs (lymphocytes) on the intrusion detection time in *IMIDG* and the current IDS

### 4 Conclusions

The proposed immunity and mobile agent based intrusion detection for grid (*IMIDG*) model is an active grid security technique.

In *IMIDG*, the concept of *self*, *nonself*, *antigen*, *antibody*, *agent* and *match algorithm* have been abstracted and extended. Besides, *IMIDG* introduces a new idea of *dynamic memory MoAs survival*, which enhances detection efficiency and assures steady performance in immune-based IDS. Furthermore, the mathematical models of *self*, *mature MoA*, *dynamic memory MoA survival*, *CoA* and *BoA* are presented.



The effects of several import parameters on system performance and detection efficiency in the model of dynamic memory MoA survival are analyzed and shown in the experiments. The theoretical analysis and the experiment results show that the proposed method is an effective solution to intrusion detection for grid security.

## 5 Future Researches

The effects of several import parameters on system performance are very important in the models of mature MoA and dynamic memory MoA survival. The set of appropriate parameters could enhance detection efficiency and assure steady performance.

These parameters include the new age  $T$  and count  $\beta-1$  of degraded memory cells in the models of mature MoA,  $K$  and  $P$  in the models of dynamic memory MoA survival.

## Acknowledgment

This work is partially supported by the National Natural Science Foundation of China under Grant No. 60373110, 60573130 and 60502011, the National Research Foundation for the Doctoral Program of Higher Education of China under Grant No.20030610003, the New Century Excellent Expert Program of Ministry of Education of China under Grant No. NCET-04-0870, and the Innovation Foundation of Sichuan University under Grant No.2004CF10.

## References

1. Foster, Carl Kesselman, Gene Tsudik and Steven Tuecke: A Security Architecture for Computational Grids. In Proc. of Computer and Communication Security (1998)
2. Fang-Yie Leu, Jia-Chun Lin, Ming-Chang Li, Chao-Tung Yang: A Performance-Based Grid Intrusion Detection System. In Proc. of International Computer Software and Applications, Vol.1 (2005) 525-530
3. M. Tolba, M. Abdel-Wahab, I. Taha, and A. Al-Shishtawy: GIDA: Toward Enabling Grid Intrusion Detection Systems. In Proc. of the Second International Intelligent Computing and Information Systems Conference (2005)
4. S. Forrest, S. Hofmeyr, and A. Somayaji: Computer Immunology. In Proc. of Communications of the ACM, Vol. 40 (1997) 88-96
5. A. Somayaji, S. A. Hofmeyr, and S. Forrest: Principles of a Computer Immune System. In Proc. of the New Security Paradigms'97 (1997) 75-82
6. Tao Li: Compute immunology. Publishing House of Electronics Industry, Beijing (2004)
7. T. Li: A New Model for Dynamic Intrusion Detection. Lecture Notes in Computer Science. Springer-Verlag, Berlin Heidelberg New York (2005) 72 - 84
8. T. Li: An immune based dynamic intrusion detection model. Chinese Science Bulletin (2005) 2650 - 2657
9. T. Li: An immunity based network security risk estimation. Science in China Ser. F Information Sciences (2005) 557 - 578

10. T. Li: An Immune-Based Model for Computer Virus Detection. Lecture Notes in Computer Science. Springer-Verlag, Berlin Heidelberg New York (2005) 59 - 71
11. D. Dasgupta: Immunity-based intrusion detection system: A general framework. In Proc. of 22nd National Information Systems Security Conference (1999) 147-160
12. R. B. Machado, A. Boukerche, J. B. M. Sobral, K. R. L. Juca and M. S. M. A. Notare: A Hybrid Artificial Immune and Mobile Agent Intrusion Detection Base Model for Computer Network Operations. In Proc. of the 19th IEEE International Parallel and Distributed Processing (2005)
13. Paul K. Harmer, Paul D. Williams, Gregg H. Gunsch, and Gary B. Lamont: An Artificial Immune System Architecture for Computer Security Applications. In Proc. of IEEE Transactions on Evolutionary Computation, Vol. 6 (2002)
14. M. Murshed, R. Buyya, D. Abramson: GridSim: A Grid Simulation Toolkit for Resource Management and Scheduling in Large-Scale Grid Computing Environments. In Proc. of the 17th IEEE International Symposium on Parallel and Distributed (2002)

# Description Logic Based Composition of Web Services

Fen Lin<sup>1,2</sup>, Lirong Qiu<sup>1,2</sup>, He Huang<sup>1,2</sup>, Qing Yu<sup>2</sup>, and Zhongzhi Shi<sup>1</sup>

<sup>1</sup> Key Laboratory of Intelligent Information Processing, Institute of Computing Technology, Chinese Academy of Sciences, China

<sup>2</sup> Graduate University of the Chinese Academy of Sciences  
{linf, qiulr, huangh, yuq, shizz}@ics.ict.ac.cn

**Abstract.** Automatic service composition may dramatically improve development efficiency of Web Service applications. This paper proposes an approach to automatically process semantic and dynamic service composition using Description Logics(DLs) and AI planning techniques. Services and service composition problems are formalized with DLs to provide well-defined semantics. Four relationships among services as well as two combined service expressions are defined, with which AI planning techniques can be used to reason about how to compose services to achieve user-defined goals. We present an algorithm for automatic service composition, which uses backward-chaining search of potential services, and automatically eliminates irrelevant services while selecting, thus guarantee the execution efficiency. We also make some performance optimization of the algorithm such as removing redundant services and reusing previously achieved goals. All the composition steps could be done dynamically and automatically. Finally, we present an example to show how the algorithm works.

**Keywords:** Semantic Web Services(SWS), Web Services Composition (WSC), Description Logic(DL).

## 1 Introduction

Web services provide a standard means of interoperating between different software applications, running on a variety of platforms and/or frameworks [1]. In a Web service application, if there is no single Web service that can achieve the goal required by the user, there should be a software agent which can automatically compose existing services together in order to fulfill the request. Automatic service composition has the potential to reduce development time and effort for new applications. However, it is a hard problem, and the first step toward which is to give a well-defined formal semantics of the services.

DLs are a family of knowledge representation languages that are able to represent structural knowledge in a formal and well-understood way [2], which play an important role in the Semantic Web since they are the basis of the W3C-recommended Web ontology language OWL [3,4]. However, automatic service

composition not only needs static information, but also needs dynamic information to allow the users to effect changes in the world. In this paper, we use DLs to formalize the services in order to provide a well-defined semantics based on [5], with the description of the preconditions and effects of a service. We also define four relationships among the services as well as two combined service expressions, with which we can reason about how to compose services to achieve the user-defined goals.

Recently there has been a lot of work applying AI planning techniques to the service composition problem such as [6,7]. Despite the work done, composition is still largely unexplored. The combination of DLs and AI planning techniques becomes directly relevant to service composition. In this paper, we use DLs to formalize the composition problem in a general and formal plan execution task. We also propose an algorithm for automatic service composition. The algorithm uses backward-chaining search of potential services, and automatically eliminates irrelevant services while selecting, thus guarantee the execution efficiency. We also make some performance optimization of the algorithm such as removing redundant services and reusing previously achieved goals. All the composition steps could be done dynamically and automatically.

The arrangement of the rest of this paper is as follows. Section 2 gives a brief survey of the DLs. In Section 3, we introduce the definition of the semantic web service and its formal semantics. Then we define four relationships among the services. Besides, we define two combined service expressions: sequence and parallel, which are used to composite the services. In Section 4, we present the composition algorithm in detail including: the service composition problem, removing redundant services, composition algorithm, performance evaluation and a travel example. Finally, Section 5 contains a brief conclusion and describes the future plans.

## 2 Preliminaries

This section gives a short and formal introduction to DLs. See [2,8] for a comprehensive introduction or details. DLs are a family of knowledge representation languages that are able to represent structural knowledge in a formal and well-understood way. A description logic system consists of four parts: constructors which represent concept and role, Terminological assertion (Tbox) subsumption assertion, Assertions about individual (Abox) instance assertion, and reasoning mechanism of Tbox and ABox.

The constructors determine the expressive power of the DL. Given mutually disjoint sets  $N_C$  of concept names,  $N_R$  of role names, and  $N_I$  of individual names, concept and role constructors in  $\mathcal{ALCN}\mathcal{R}$ , can be defined using the following syntax:

$$\begin{aligned} C, D &\rightarrow A \mid \top \mid \perp \mid C \sqcup D \mid C \sqcap D \mid \neg C \mid \forall R. C \mid \exists R. C \mid (\geq nR) \mid (\leq nR) \\ R &\rightarrow P_1 \sqcap \dots \sqcap P_m \end{aligned}$$

DL commonly have a set-theoretic semantics. The semantics of a DL knowledge base are given via *interpretation*  $\mathcal{I} = \langle \Delta^{\mathcal{I}}, \cdot^{\mathcal{I}} \rangle$ , where  $\Delta^{\mathcal{I}}$  is a non-empty set of objects, and  $\cdot^{\mathcal{I}}$  maps

- each individual name  $a$  to an element  $a^{\mathcal{I}} \in \Delta^{\mathcal{I}}$ ;
- each concept name  $C$  to a subset of  $\Delta^{\mathcal{I}}$ , i.e.,  $C^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$ ;
- each role name  $R$  to a binary relation on  $\Delta^{\mathcal{I}}$ , i.e.,  $R^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$ .

Let  $\mathcal{T}$  be an acyclic Tbox, and  $\mathcal{A}$  denotes an Abox. We say that  $\mathcal{I}$  satisfies  $\mathcal{T}, \mathcal{A}$ , written as  $\mathcal{I} \models \mathcal{T}, \mathcal{A}$ , if  $\mathcal{I}$  satisfies every sentence in  $\mathcal{T}, \mathcal{A}$ .

We define the binary relations  $\equiv$  and  $\preceq$  on models of  $\mathcal{I}, \mathcal{I}'$  in  $\mathcal{A}$  w.r.t.  $\mathcal{T}$ ,  $\mathcal{I} \equiv \mathcal{I}'$ , this means the two models are equal, that is to say,

$$a^{\mathcal{I}} = a^{\mathcal{I}'} \text{ and } C^{\mathcal{I}} = C^{\mathcal{I}'} \text{ and } R^{\mathcal{I}} = R^{\mathcal{I}'};$$

$\mathcal{I} \preceq \mathcal{I}'$ , this means one model is inferior to the other, that is to say,

$$a^{\mathcal{I}} = a^{\mathcal{I}'} \text{ and } C^{\mathcal{I}} \subseteq C^{\mathcal{I}'} \text{ and } R^{\mathcal{I}} \subseteq R^{\mathcal{I}'}.$$

In a DL reasoning system, several different kinds of reasoning problems are typically supported w.r.t. a knowledge base  $\mathcal{T}, \mathcal{A}$ . For the purpose of this paper, it suffices to introduce concept satisfiability and ABox consistency:

- **Concept satisfiability:** the concept  $C$  is satisfiable w.r.t. the TBox  $\mathcal{T}$  iff there exists a model  $\mathcal{I}$  of  $\mathcal{T}$  such that  $C_{\mathcal{I}} \neq \emptyset$
- **ABox consistency:** the ABox  $\mathcal{A}$  is consistent w.r.t. the TBox  $\mathcal{T}$  iff there exists an interpretation  $\mathcal{I}$  that is a model of both  $\mathcal{T}$  and  $\mathcal{A}$ .

All other inferential services can be realized with satisfiability tests of knowledge bases. For example, the problem whether  $\mathcal{I} \models C \sqsubseteq D$  holds, is equivalent to the problem whether  $\mathcal{I} \cup \{a : C, a : \neg D\}$  is satisfiable(consistent).

### 3 Service Descriptions

In this section, to begin with we introduce the definition of Semantic Web services and its formal semantics based on [9]. Then we define several relationships among them, which are useful to remove redundant services in the following section. Furthermore, we define two combined service expressions: sequence and parallel, which are used to composite the services to achieve the user-defined goal. Finally, we present some theorems about these operators.

#### 3.1 Service Definition

**Definition 1 (Semantic Web Services).** Let  $\mathcal{T}$  be an acyclic Tbox. An atomic Semantic Web Service  $S$  for  $\mathcal{T}$  is defined in the form of  $S = (P, E)$ , where:

- $S$  is the service name.
- $P$  is a finite set of Abox assertions, the pre-conditions, which must be satisfied before the service is executed.
- $E$  is a finite set of conditional post-conditions, which denote the effects of the service,  $E$  is a set of pair  $\varphi/\psi$ , where  $\varphi$  is an Abox assertions for  $\mathcal{T}$ ,  $\psi$  is a primitive literal for  $\mathcal{T}$ , i.e.,  $C(p), \neg C(p), R(p, q), \neg R(p, q)$  with  $C$  a primitive concept in  $\mathcal{T}$  and  $R$  a role name.

Each Semantic Web Service can be specified by its preconditions and effects in the planning context. Precondition presents logical conditions that should be satisfied prior to the service being requested. Effects are the result of the successful execution of the service. To illustrate the definition of services, consider a travel website, whose business is providing services for tourists including traffic service, hotel service, and taxi service as defined below.

- $TrafficService = (\{ Person(a) \}, \{ bookTicket(a, b), Ticket(b) \})$
- $ModestHotelService = (\{ Person(a), \exists hasTelepone.Telephone(a) \}, \{ bookHotel(a, c), ModestHotel(c) \})$
- $PalaceHotelService = (\{ Person(a), \exists hasMoney.EnoughMoney(a) \}, \{ bookHotel(a, d), PalaceHotel(d) \})$
- $TaxiService = (\{ Person(a), \exists bookTicket.Ticket(a) \}, \{ bookTaxi(a, e), \exists bookHotel.ModestHotel(a)/ModestTaxi(e), \exists bookHotel.PalaceHotel(a)/PalaceTaxi(e) \})$

The traffic service provides ticket booking for the customer and everybody can book a ticket for himself. The modest hotel service provides a modest hotel booking for the customer by telephone. The palace hotel service provides a palace hotel booking for the customer if the customer has a great amount of money. Which hotel to choose is decided by the economic ability of the customer. The taxi service provides the transportation for the daily routines of the customer if the customer has booked the ticket. Which taxi to choose is decided by the choice of the hotel, if the customer has booked a modest hotel, it will book a modest taxi, otherwise a palace taxi.

The meaning of the concepts used in these services are defined in the following acyclic Tbox  $T$ :  $T = \{$

$$\begin{aligned} EnoughMoney &\equiv Money \sqcap \geq 10 \text{ hasThousandRMB}, \\ Hotel &\equiv ModestHotel \sqcup PalaceHotel, \\ Telephone &\equiv FixedLinePhone \sqcup MobilePhone, \\ Taxi &\equiv ModestTaxi \sqcup PalaceTaxi. \end{aligned}$$

The formal semantics of services can be defined by means of a transition relation on interpretations. The service  $S$  may transform  $\mathcal{I}$  to  $\mathcal{I}' (\mathcal{I} \Rightarrow_S^{\mathcal{T}, \mathcal{A}} \mathcal{I}')$ , if  $C$  is a primitive concept and  $R$  a role name, then

- $C^{\mathcal{I}'}$ :  $= (C^{\mathcal{I}} \cup \{c^{\mathcal{I}} | \varphi/C(c) \in E, \text{ and } \mathcal{I} \models \varphi\} \setminus \{c^{\mathcal{I}} | \varphi/\neg C(c) \in E, \text{ and } \mathcal{I} \models \varphi\})$
- $R^{\mathcal{I}'}$ :  $= (R^{\mathcal{I}} \cup \{(a^{\mathcal{I}}, b^{\mathcal{I}}) | \varphi/R(a, b) \in E, \text{ and } \mathcal{I} \models \varphi\} \setminus \{(a^{\mathcal{I}}, b^{\mathcal{I}}) | \varphi/\neg R(a, b) \in E, \text{ and } \mathcal{I} \models \varphi\})$

Then we present executability and projection of services as follows [9]:

- **Executability:**  $S$  is executable in  $\mathcal{A}$  w.r.t.  $\mathcal{T}$  iff  $\mathcal{I} \models P$  in all models  $\mathcal{I}$  of  $\mathcal{A}$  w.r.t.  $\mathcal{T}$  and  $\mathcal{I} \Rightarrow_{\mathcal{S}}^{\mathcal{T}} \mathcal{I}', \mathcal{I}'$  with no conflict.
- **Projection:** an assertion  $a$  is a consequence of applying  $S$  in  $\mathcal{A}$  w.r.t.  $\mathcal{T}$  iff for all models  $\mathcal{I}$  of  $\mathcal{A}$  and  $\mathcal{T}$ , and all  $\mathcal{I}'$  with  $\mathcal{I} \Rightarrow_{\mathcal{S}}^{\mathcal{T}} \mathcal{I}'$ , we have  $\mathcal{I}' \models a$ .

### 3.2 Service Relationship

Among semantic web services, there exist relationships, which are the important features of services for composition. Let  $\mathcal{T}$  be an acyclic Tbox, and  $\mathcal{A}$  be an Abox, and  $S_i$  denotes services. Formally, we define the following relationships about services.

- **Identical Service:**  $S_1 = S_2$ , this means the two services can provide the same function in spite of the fact that they may have different service names.
- **Conditionally Identical Service:**  $S_1 \cong S_2$ , this means  $S_1$  can provide the same function as  $S_2$  in some situation. For instance, there is a traffic service that provides ticket booking just for students,  $StudentTrafficService = (\{Student(a)\}, \{bookTicket(a, b), Ticket(b)\})$ , and add  $Student \sqsubseteq Person$  to  $Tbox(\mathcal{T})$ . If all the people in the system are students, then we have that,  $StudentTrafficService \cong TrafficService$  in the system.
- **Substitute Service:**  $S_1 \triangleleft S_2$  or  $S_2 \triangleright S_1$ , this means a service  $S_1$  can be substituted by service  $S_2$  in any case, if  $S_1$  is possible,  $S_2$  is also possible and has exactly the same effects as  $S_1$  or more. For instance, as described above,  $StudentTrafficService$  is a sub-service of  $TrafficService$ . Of course, the former ticket service can be alternated by the other service.
- **Conditionally Substitute Service:**  $S_1 \trianglelefteq S_2$  or  $S_2 \trianglerighteq S_1$ , this means a service  $S_1$  can be substituted by service  $S_2$  in some situation. For instance, there is a traffic service that provides train ticket booking for everybody,  $TrainTrafficService = (\{Person(a)\}, \{bookTicket(a, b), TrainTicket(b)\})$ , and add  $Ticket \equiv PlaneTicket \sqcup TrainTicket$  to  $Tbox(\mathcal{T})$ . If all the people in the system are students, then we have that,  $TrainTrafficService \trianglelefteq StudentTrafficService$  in the system.

Obviously, we have the following theorems about the relationships on services.

**Theorem 1.** *If  $S_1, S_2$  are executable in  $\mathcal{A}$  w.r.t.  $\mathcal{T}$ , then  $\mathcal{I} \Rightarrow_{S_1}^{\mathcal{T}, \mathcal{A}} \mathcal{I}', \mathcal{I} \Rightarrow_{S_2}^{\mathcal{T}, \mathcal{A}} \mathcal{I}''$ , in all models  $\mathcal{I}$  of  $\mathcal{A}$  w.r.t.  $\mathcal{T}$ , if we have  $\mathcal{I}' \equiv \mathcal{I}''$  hold, then  $S_1 \cong S_2$  in the case of  $\mathcal{A}$  w.r.t.  $\mathcal{T}$ .*

**Theorem 2.** *If  $S_1, S_2$  are executable in  $\mathcal{A}$  w.r.t.  $\mathcal{T}$ , then  $\mathcal{I} \Rightarrow_{S_1}^{\mathcal{T}, \mathcal{A}} \mathcal{I}', \mathcal{I} \Rightarrow_{S_2}^{\mathcal{T}, \mathcal{A}} \mathcal{I}''$ , in all models  $\mathcal{I}$  of  $\mathcal{A}$  w.r.t.  $\mathcal{T}$ , if we have  $\mathcal{I}' \ll \mathcal{I}''$  hold, then  $S_1 \trianglelefteq S_2$  in the case of  $\mathcal{A}$  w.r.t.  $\mathcal{T}$ .*

### 3.3 Combined Service

A combined service is an aggregation of some independent and interactive web services (individual or combined web services). We define two combined

service expressions: sequence and parallel as follows. Let  $\mathcal{T}$  be an acyclic Tbox,  $\mathcal{A}$  denotes an Abox and  $S_i$  denotes services for  $\mathcal{T}$ .

**Definition 2 (Sequence Services).**  $S = S_1; S_2; \dots; S_k$ . A sequence service  $S$  is a service that is achieved by an order of services in a sequence, a following of one service after another.

- **Executability:**  $S = S_1; S_2; \dots; S_k$  is executable in  $\mathcal{A}$  w.r.t.  $\mathcal{T}$  iff the following conditions are true in all models  $\mathcal{I}$  of  $\mathcal{A}$  and  $\mathcal{T}$  :
  - $\mathcal{I} \models P_1$  and
  - For all  $i$  with  $1 \leq i < k$  and all interpretations  $\mathcal{I}'$  with  $\mathcal{I} \Rightarrow_{S_1; S_2; \dots; S_i}^{\mathcal{T}} \mathcal{I}'$ , we have  $\mathcal{I}' \models P_{(i+1)}$  and
  - $\mathcal{I} \Rightarrow_S^{\mathcal{T}} \mathcal{I}'$ ,  $\mathcal{I}'$  with no conflict.

**Definition 3 (Parallel Services).**  $S = S_1|S_2|\dots|S_k$ . A parallel service  $S$  is a service that is achieved by a set of services of equal rank, all services executing at the same time.

- **Executability:**  $S = S_1|S_2|\dots|S_k$  is executable in  $\mathcal{A}$  w.r.t.  $\mathcal{T}$  iff each service in  $S$  is executable.

A combined service can be composed by sequence and parallel services. For example, we can compose a combined service of the foregoing services such as  $TravelService = (TrafficService|HotelServices); TaxiServices$ . By executing the travel service, the customer will have the suitable ticket, hotel and taxi for his trip.

After the definitions, we have some theorems of the operators as follows, which is used to reduce the complexity of the combined services.

**Theorem 3.**  $S_1 \trianglelefteq (S_1; S_2)$  in the case of  $\mathcal{A}$  w.r.t.  $\mathcal{T}$ , if  $S_1; S_2$  are executable in  $\mathcal{A}$  w.r.t.  $\mathcal{T}$ .

*Proof.* If  $S_1$  is possible,  $S_1; S_2$  is also possible, and all the effects  $S_1$  produced obviously can be produced by  $S_1; S_2$ .

**Theorem 4.**  $S_1|S_2 = S_2$  iff  $S_1 \trianglelefteq S_2$  in the case of  $\mathcal{A}$  w.r.t.  $\mathcal{T}$ , if  $S_1|S_2$  are executable in  $\mathcal{A}$  w.r.t.  $\mathcal{T}$ .

*Proof.* If  $S_1$  can be substituted by  $S_2$ , executing  $S_1|S_2$  has the same effects as executing  $S_2$ , or vice versa.

**Theorem 5.**  $S_1|S_2|S_3 = (S_1|S_2)|S_3 = S_1|(S_2|S_3)$ , that is to say, the parallel services are irrelevant with the order of the services.

## 4 Planning for Service Composition

With the results of Section 2 and Section 3, we have addressed a fundamental barrier to automated service composition. This section aims to give a specific approach to automated composition of web services. Most important of all we



use DLs to formalize the composition problem and present its formal definition. Moreover we remove the redundant services, which is desirable because it reduces the plan search space. After the preprocessing, we propose the planning algorithm with an analysis of the algorithm. Finally we provide an example to show how the algorithm works.

#### 4.1 Service Composition Problem

A Web services composition problem can be described as: Given a set of web services and a description of some tasks or goals to be achieved (e.g., "Make the travel arrangements for my trip"), find a composition of services that achieves the task. And it can be viewed as a planning problem, depending upon how we represent our services. In this paper, we use DLs to formalize the services as described in Section 3. Then the definition of the composition problem is as follows:

**Definition 4 (Service Composition Problem).** *the composition problem can be described as a four-tuple  $\langle \mathcal{T}, \mathcal{A}, \mathcal{G}, \mathcal{S} \rangle$ , where:*

- $\mathcal{T}$  describes the vocabulary of the application domain.
- $\mathcal{A}$  contains assertions about named individuals in terms of this vocabulary and also denotes the initial state of the world.
- $\mathcal{G}$  is a set of assertions, which represent the goal attempting to reach.
- $\mathcal{S}$  is the set of services as described in Section 3.1.

*A composition problem is to find a combined service performed to reach the goal.*

#### 4.2 Removing Redundant Services

**Removing Weaker Services:** In section 3.2, we define substitute relationship among services. If a service  $S_1$  can be substituted by service  $S_2$  in any case,  $S_1$  may be redundant in the sense that in any situation, if  $S_1$  is possible,  $S_2$  is also possible and has exactly the same effects as  $S_1$ . More generally, we define the notion that service  $S_2$  is stronger than service  $S_1$ . So we can remove service  $S_1$  to reduce the plan search space. Each weaker services, which can be substituted by another service, could be removed. The pseudo-code for removing weaker services is in Algorithm 1.

**Removing Useless Services:** If the preconditions of the service  $S$  can never be satisfied in any case, then  $S$  is useless and can be removable. The pseudo-code for removing useless services is in Algorithm 2. For simplicity, we just consider those services whose preconditions is inconsistent with  $\mathcal{G}$  in  $\mathcal{A}$  w.r.t.  $\mathcal{T}$ . These services are not possible, otherwise there will be a conflict in the system.

**Removing irrelevant Services with respect to a Goal:** If the effects of the service  $S$  can never make the goal true and if  $S$  can not directly achieve the preconditions of any services, then  $S$  is irrelevant with respect to the goal and can be removed. The composition algorithm described in Section 4.3 is goal-driven, thus the services irrelevant to the goal are eliminated automatically while selecting.

---

**Algorithm 1.** Removing Weaker Services

---

```

1:  $\mathcal{T}$  is the Tbox
2:  $\mathcal{A}$  is the ABox
3:  $\mathcal{S} = (S_1, \dots, S_n)$ 
4: for  $i = 1$  to  $n$  do
5:   if there exists  $S_k \in \mathcal{S}, k! = i$ , and  $S_i \sqsubseteq S_k$  in the case of  $A$  w.r.t.  $T$  then
6:     remove  $S_i$  from  $\mathcal{S}$ 
7:   end if
8: end for

```

---



---

**Algorithm 2.** Removing useless Services

---

```

1:  $\mathcal{T}$  is the Tbox
2:  $\mathcal{A}$  is the ABox
3:  $\mathcal{G}$  is the Goal
4:  $\mathcal{S} = (S_1, \dots, S_n)$ 
5: for  $i = 1$  to  $n$  do
6:    $S_i = (P_i, E_i)$ 
7:   if  $P_i$  is inconsistent with  $\mathcal{G}$  in  $\mathcal{A}$  w.r.t.  $\mathcal{T}$ . then
8:     remove  $S_i$  from  $\mathcal{S}$ 
9:   end if
10: end for

```

---

### 4.3 Composition Algorithm

The algorithm is a simple backward-chaining algorithm, which executes as follows. To begin with the user initializes the algorithm by specifying the goals. Then the algorithm divides the goals into two parts: a sub goal(*subGoal*) and the rest goals(*restGoal*). Furthermore the algorithm checks if the *subGoal* has already been satisfied previously. If so, the algorithm includes *subServices* in the solution. Now the algorithm tries to satisfy the rest goals. This corresponds to a new iteration which involves executing the same steps and include *restServices* in the solution, thus return *subServices;restServices*. If not, the algorithm checks if the *subGoal* could be satisfied by the initial assertions. If so, the algorithm tries to satisfy the rest goals as described above and return *restServices*. If not, the algorithm looks for service(s) whose effect(s) could satisfy the sub-goal and include *subServices* in the solution. Now the preconditions of the newly included service(s) are treated as unsatisfied goals. And the algorithm tries to satisfy the unsatisfied goals. This corresponds to a new iteration and includes *preServices* in the solution. And then the algorithm tries to satisfy the rest goals as described above and include *restServices* in the solution, thus return  $(preServices; subServices)|restServices$ . Note that the algorithm terminates when failing to find any services that satisfy any of the unsatisfied goals. If all unsatisfied goals correspond to the initial assertions then the algorithm could be successful. The pseudo-code for automatic service composition is in Algorithm 3.

**Algorithm 3**


---

```

1:  $\mathcal{T}$  is the Tbox
2:  $\mathcal{A}$  is the ABox, add the initial state to  $\mathcal{A}$ 
3:  $\mathcal{G}$  is the Goal
4:  $\mathcal{S} = (S_1, \dots, S_n)$ 
5:  $GoalServices = \emptyset$ 
6: if  $\mathcal{T}, \mathcal{A} \models \mathcal{G}$  then
7:    $GoalServices = \emptyset$ 
8:   return  $GoalServices$ 
9: end if
10: select an assertion  $subGoal$  from the unsatisfied goal  $\mathcal{G}$ 
11:  $restGoal = \mathcal{G} - subGoal$ 
12: if the  $subGoal$  has already been processed by  $subServices$  then
13:   executed recursively, and get  $restServices$  to reach  $restGoal$ .
14:    $GoalServices = subServices|restServices$ 
15:   return  $GoalServices$ 
16: else if  $\mathcal{T}, \mathcal{A} \models subGoal$  then
17:   executed recursively, and get  $restServices$  to reach  $restGoal$ .
18:    $GoalServices = restServices$ 
19:   return  $GoalServices$ 
20: else if there exists no services to satisfy the subgoal  $subGoal$  then
21:   return  $NULL$ 
22: else
23:   there exists a set of services  $\mathcal{S}' = \{S'_1, \dots, S'_k\}$  can satisfy the subgoal
24:   loop
25:     select a set of services  $subServices$  from  $\mathcal{S}'$  that can satisfy the subgoal
26:     there exists a set of preconditions to satisfy the subgoal,  $\mathcal{G}' = \{G_1, \dots, G_m\}$ ,
       where  $G_i = Pre(subServices) + Post_i(subServices, subGoal)$ 
27:     loop
28:       select a  $preGoal$  from  $\mathcal{G}'$  that can satisfy the subgoal
29:       executed recursively, and get  $preServices$  to reach  $preGoal$ .
30:       executed recursively, and get  $restServices$  to reach  $restGoal$ .
31:       if  $preServices! = NULL$  then
32:         if  $restServices! = NULL$  then
33:            $GoalServices = (preServices; subServices)|restServices$ 
34:           return  $GoalServices$ 
35:         end if
36:       end if
37:       remove  $preGoal$  from  $\mathcal{G}'$ 
38:     end loop
39:     remove  $subServices$  from  $\mathcal{S}'$ 
40:   end loop
41:   return  $NULL$ 
42: end if

```

---

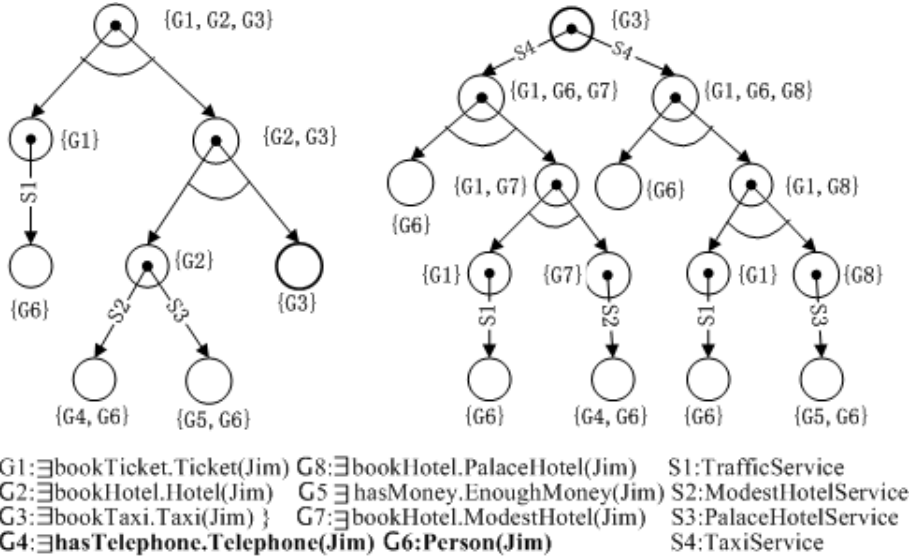


Fig. 1. Example of Service Composition Algorithm Execution Tree

#### 4.4 Performance Evaluation

In the service composition problem, we make a closed world assumption, and the solution either exists or does not exist. If the solution exists, the algorithm will surely find one and return it. If not, the algorithm will return NULL. All the composition steps could be done dynamically and automatically.

Most important of all we remove the redundant services including weaker services and useless services, which is desirable because it reduces the search space and improves efficiency in some cases. Note that removal of weaker primitive actions may result in removal of the optimal plan. We may lose the optimal plan with respect to the number of primitive actions in our initial domain. The search tree created by the above algorithm can be seen as a sort of AND-OR tree, where the "OR" branches represent different ways of satisfying the goal, and the "AND" branches represent combinations of service effects that together reach the goal. Leaves in the tree represent initial assertions to be available. Viewing the search and execution as an AND-OR tree allows us to investigate the use of different search techniques to improve execution performance. Moreover the algorithm described above uses backward-chaining search of potential services, and automatically eliminates irrelevant services while selecting, thus guarantee the execution efficiency. Finally the algorithm reuses previously achieved goals, which makes the algorithm more efficient.

#### 4.5 A Sample Case

In this section we provide an example that illustrates how the algorithm works. For example, a tourist named Jim wants to have a trip to Beijing, and he wants

the website to make the travel arrangements for his trip. We suppose that the tourist has a telephone but not enough money. The problem could be formalized as  $\langle \mathcal{T}, \mathcal{A}, \mathcal{G}, \mathcal{S} \rangle$ , where:

- $\mathcal{T}, \mathcal{S}$  defined in Section 3.1.
- $\mathcal{A} = \{Person(Jim), hasTelephone(Jim, t), MobilePhone(t)\}$ .
- $\mathcal{G} = \{\exists bookTicket.Ticket(Jim), \exists bookHotel.Hotel(Jim), \exists bookTaxi.Taxi(Jim)\}$ .

The algorithm executes as the Figure 1 shown. To start with we divide the goals  $\{G1, G2, G3\}$  into two parts:  $\{G1\}$  and  $\{G2, G3\}$ , as shown in the left tree in Figure 1. And there exists  $S1$  to satisfy the first subgoal. Next we try to satisfy the rest goals, which leads to a new iteration. We find that  $G2$  could be satisfied by  $S2$  and  $S3$ . Obversely, the preconditions of the former service could be satisfiable in  $\mathcal{A}$  w.r.t.  $\mathcal{T}$ , while the other couldn't. In addition we try to reach the rest goal,  $\{G3\}$ , as shown in the right tree in Figure 1. There exist only one service to meet the requirement. And we have two *preGoals* of the service:  $\{G1, G6, G7\}$  and  $\{G1, G6, G8\}$ , each leads to a new iteration. Obviously, the subgoal,  $\{G6\}$ , is in the initial assertions. Now the left goals are treated as the unsatisfied goals. The subgoal  $\{G1\}$  has already been satisfied and could be achieved by  $S1$ . the goals  $G7$  and  $G8$  could be achieved by  $S2$  and  $S3$  respectively. The *preGoals* :  $\{G4, G6\}$  can be satisfiable, but the other  $\{G5, G6\}$  couldn't. Finally we get the solution,  $(S1|S2)|((S1|S2); S4)$ , which could be reduced to  $(S1|S2); S4$  by Theorem 3 and 4. So the result service of the problem is  $(TrafficService|ModestHotelService); TaxiService$ .

## 5 Conclusions and Future Work

In this paper, semantic web services and their composition are studied and a specific approach to web services composition is proposed. In the approach, we use DLs to formalize the services and the services composition problem in order to provide a well-defined semantics. And we define four relationships among the services as well as two combined service expressions including sequence and parallel, with which AI planning techniques can be used to reason about how to compose services to reach the user-defined goal. The algorithm uses a backward-chaining search with some performance optimizing including removing redundant services and reusing previously achieved goals. All the composition steps could be done dynamically and automatically.

The idea presented in this paper can be extended in future from different points of view. One is the improvement of the composition algorithm. If there exist various composition solutions to achieve the user-defined goal, which one should be chosen to satisfy the goal best? Another is the improvement of the composition model. If the goals couldn't be satisfied, how can we find the largest compatible subset of the goals to achieve. Another is the implementation of the algorithm. From a multi-agent systems perspective, a service composition software system can be viewed as a collection of sociable agents, representing

individual services, which cooperate, coordinate and work together to achieve the goal [10]. So we can implement the system on MAGE [11], which is a Multi-AGENT Environment developed by Intelligent Science Research Group, at Key Lab of IIP, ICT, CAS, China, using agent technology to analyze, design, implement and deploy multi-agent systems.

## Acknowledgements

Our work is supported by the Natural National Science Foundation of China (No.60435010), the National 973 Project of China (No.2003CB317004) and the Natural Science Foundation of Beijing (No.4052025).

## References

1. David Booth et al. Web services architecture. Technical report, W3C Working Group Note, 2004. See <http://www.w3.org/TR/ws-arch/>.
2. Franz Baader, Diego Calvanese, Deborah McGuinness, Daniele Nardi, and Peter F. Patel-Schneider. *The Description Logic Handbook: Theory, Implementation, and Applications*. Cambridge University Press, 2003.
3. Franz Baader, Ian Horrocks, and Ulrike Sattler. Description logics as ontology languages for the semantic web. In *In Festschrift in honor of Jorg Siekmann, LNAI*, Springer-Verlag, 2005.
4. Ian Horrocks, Patel-Schneider, and Frank van Harmelen. From shiq and rdf to owl: The making of a web ontology language. *Journal Web Semantics*, 1(1):7–26, 2003.
5. F. Baader, M. Milicic, C. Lutz, U. Sattler, and F. Wolter. Integrating description logics and action formalisms for reasoning about web services. LTCS-Report LTCS-05-02, Chair for Automata Theory, Institute for Theoretical Computer Science, Dresden University of Technology, Germany, 2005. See <http://lat.inf.tu-dresden.de/research/reports.html>.
6. Marco Aiello, Mike P. Papazoglou, Jian Yang, M. Carman, Marco Pistore, Luciano Serafini, and Paolo Traverso. A request language for web-services based on planning and constraint satisfaction. In *Proceedings of the Third International Workshop on Technologies for E-Services*, pages 76–85, Springer-Verlag London, UK, 2002.
7. S. McIlraith and T. Son. Adapting golog for composition of semantic web services. In *Proceedings of the 8th International Conference on Principles of Knowledge Representation and Reasoning (KR 2002)*, pages 482–493, 2002.
8. Description Logics website <http://dl.kr.org>.
9. F. Baader, C. Lutz, M. Milicic, U. Sattler, and F. Wolter. A description logic based approach to reasoning about web services. In *Proceedings of the WWW 2005 Workshop on Web Service Semantics (WSS2005)*, Chiba City, Japan, 2005.
10. Paul A. Buhler and José M. Vidal. Toward the synthesis of web services and agent behaviors. In *Proceedings of the First International Workshop on Challenges in Open Agent Systems*, pages 25–29, 2002.
11. Zhongzhi Shi, Haijun Zhang, and Mingkai Dong. Mage: Multi-agent environment. In *ICCNMC-03*, pages 181–188, 2003.

# A Reputation Multi-agent System in Semantic Web

Wei Wang, Guosun Zeng, and Lulai Yuan

Department of computer Science and Technology, Tongji University,  
Shanghai 201804, China  
Tongji Branch, National Engineering & Technology Center of High Performance Computer,  
Shanghai 201804, China  
willtongji@gmail.com

**Abstract.** Though research on the Semantic Web has progressed at a steady pace, its promise has yet to be realized. One major difficulty is that, by its very nature, the Semantic Web is a large, uncensored system to which anyone may contribute, especially when using a P2P-based multi-agent infrastructure for knowledge sharing. This raises the question of how much credence to give each information source. We cannot expect each user to know the trustworthiness of each source. We tackle this problem by employing a reputation mechanism, which offer a viable solution to encouraging trustworthy behavior in Semantic Web. In addition, we introduce a reputation multi-agent system, SemTrust, which enable Semantic Web to utilize reputation mechanism based on semantic similarity between agents. Our experiments show that the system with SemTrust outperforms the system without it and our approach is more robust with security under conditions where existing malicious agents. We hope that these methods will help move the Semantic Web closer to fulfilling its promise by using reputation-based multi-agent technology.

## 1 Introduction

The goal of Semantic Web is to build a web of meaning. Semantic Web will consist of a distributed environment of shared and interoperable ontologies, which have emerged as common formalisms for knowledge representation. The philosophy behind the Semantic Web is the same as that behind the World-Wide Web – anyone can be an information producer or consume anyone else’s information. So, it is promising to combine the P2P-based multi-agent system with Semantic Web technology. On one hand, P2P-based multi-agent system can help semantic web with sharing knowledge; on the other hand, P2P-based multi-agent system uses the semantic concept to query routing and efficient content-based location. However, even after these are in wide use, we still need to address the major issue of how to decide how trustworthy each information source is. In order to use it well, it should build a trust environment in Semantic Web.

Our method is to introduce reputation mechanism in Semantic Web to solve the problem of lack trust by using multi-agent system. We propose a reputation multi-agent system, SemTrust, to take the reputation information that is locally generated as

a result of an interaction between agents, and spread it throughout the network to produce a global reputation based on agents' semantic similarity.

The rest of this paper is organized as follows. We review some related work in Section 2. Section 3 introduces SemTrust. First describes the overview of the system model, and then we propose the local and global algorithm to compute trust value based on semantic similarity between agents. We evaluate our approach in Section 4 and analyze the experiments results. Section 5 concludes the paper.

## 2 Related Work

P2P-based multi-agent system is the sharing of computer resources and services by direct exchange between the systems. These resources and services include the exchange of information, processing cycles, cache storage, and disk storage for files. The features in P2P that makes it desirable to be an infrastructure for knowledge sharing in Semantic Web include the following:

- It encourages distributed architecture, and supports decentralization of control and access to information and services.
- It provides access to semantic information published by several independent content providers, and enables creation of personalized semantic relationships.
- It supports for publishing peer definitions and relationships to other peers and software agents.

So it is promising to use P2P-based multi-agent infrastructure in Semantic Web [1]. Ernst [8] uses P2P-based infrastructure to support the semantic web to link semantic definitions. Arumugam et al. [6] propose P2P Semantic Web (PSW) which is a platform for creating, exchanging and sharing of knowledge so that users can obtain more useful information. The key issues of their work include knowledge discovery based on semantic searching for relevant ontologies, and a peer-to-peer infrastructure in realizing ontology sharing and dissemination of knowledge on the Semantic Web. Broekstra et al. [7] propose Semantic Web and Peer-to-Peer (SWAP) project to combine Semantic Web and P2P, which takes on the challenges brought up by the novel combination of ontologies and P2P computing such that knowledge finding and sharing is effectively possible.

On the other hand, Reputation-based trust management or Reputation system [2] has been identified in the literature as a viable solution to the problem of trust in multi-agent system [9] [15] and P2P networks [10] [11] as well. The main goal of the reputation mechanism is to take the reputation information that is locally generated as a result of an interaction between agents, and spread it throughout the network to produce a global reputation. Various reputation management mechanisms have been developed. Jiang et al. [9] presents a trust construction model to achieve the trust management autonomy in multi-agent systems. Gupta et al. [10] present a partially centralized mechanism using reputation computation agents and data encryption. Xiong and Liu [11] present a feedback-based reputation mechanism where a scalar trust value for a peer is computed based on a figure of the satisfaction received by other peers. Despotovic et al.'s work documents them particularly well [3].



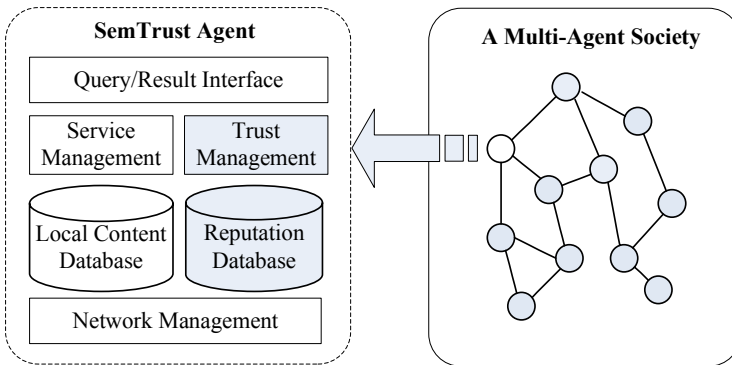
In this paper, we propose a reputation multi-agent system to utilize reputation mechanism in Semantic Web, which can take the advantage of both P2P-based multi-agent infrastructure and reputation system.

### 3 SemTrust: A Reputation Multi-agent System

#### 3.1 Overview of the Architecture Model of SemTrust

Based on the past research on knowledge sharing in a multi-ontology environment and reputation mechanism in P2P-based multi-agent system, our SemTrust approach may be implemented for any unstructured multi-agent system. For evaluation purposes, though, we use the SWAP infrastructure [7] as the foundation of the system. We recall that it provides all standard peer-to-peer functionality such as information sharing, searching and publishing of resources.

Figure 1 shows the basic building blocks of our architecture. In a multi-agent society, we assume that each agent provides a unique identifier. Similar to file sharing networks each agent may publish all resources from its local content database, so other agents can discover them by their requests. All information is wrapped as RDF statements and stored in an RDF repository. Additionally to local data each resource is assigned a topic and hierarchical information about the topics is stored. The topics an agent stores resources for are subsequently referred to as the agent's own topics.



**Fig. 1.** Overview of the Architecture Model of SemTrust

The main component is the trust management with the support of the reputation database which stores reputation data of the agents. The main goal of the reputation mechanism is to take the reputation information that is locally generated as a result of an interaction between agents, and spread it throughout the network to produce a global reputation rating for the network nodes.

In our model, the underlying agent communication component serves as a transport layer for other layers of the system model and hides all low-level communication details from the rest component. These functions may include agent discovery, agent routing, message communicating, monitoring, etc.

### 3.2 Semantic Reputation Model

We introduce a reputation model offer a viable solution to encouraging trustworthy behavior in Semantic Web. The key presumptions are that the participants of an online community engage in repeated interactions and that the information about their past doings is indicative of their future performance and as such will influence it.

Consider a multi-agent system in which the agents engage in bilateral interactions in a specific context. Assume that in each interaction a content provider and a content consumer can be identified and that the service consumer rates the provider’s trustworthiness in the interaction. The feedback set  $W$  as well as the semantics associated with its individual elements are assumed to be universally known and agreed upon. Fig. 2 presents a *trust network* example.

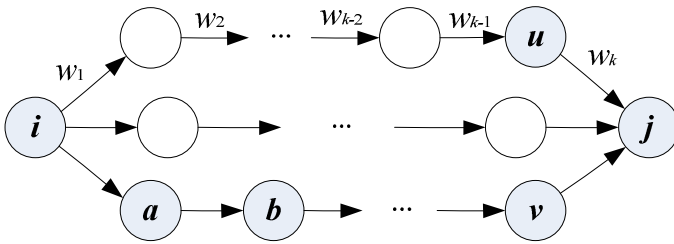


Fig. 2. Agent-based trust network

We define the feedback set  $W$  as two element set  $\{0, 1\}$ , which present binary events, and “1” means an agent is satisfies with the service provider and “0” means unsatisfied. The mathematical analysis leading to the expression for posteriori probability estimates of binary events can be found in many text books on probability theory, e.g. Papoulis & Pillai [16] p.80, and we will only present the result here.

It can be shown that posteriori probabilities of binary events can be represented by the beta distribution. The beta-family of density functions is a continuous family of functions indexed by the two parameters  $\alpha$  and  $\beta$ . The beta( $\alpha, \beta$ ) distribution can be expressed using the gamma function  $\Gamma$  as:

$$f(p | \alpha, \beta) = \frac{\Gamma(\alpha + \beta)}{\Gamma(\alpha) \cdot \Gamma(\beta)} \cdot p^{\alpha-1} (1 - p)^{\beta-1} \tag{1}$$

where  $0 \leq p \leq 1, \alpha > 0, \beta > 0$ .

with the restriction that the probability variable  $p \neq 0$  if  $\alpha < 1$ , and  $p \neq 1$  if  $\beta < 1$ .

By using Bayesian method, we define *local reputation value*  $C_{ab}$  as the sum of the feedback of the individual transactions that agent  $a$  has interacted with agent  $b$ ; equivalently, each agent  $a$  can store the number satisfactory transactions it has had with agent  $b$ ,  $u$ , and the number of unsatisfactory transactions it has had with agent  $b$ ,  $v$ . We assume that the posteriori probabilities between two agents  $a$  and  $b$  before transaction is  $\alpha = 1$  and  $\beta = 1$ . After several transactions, the distribution can be regarded as  $\alpha = u + 1$  and  $\beta = v + 1$ . So, the expectation value of the beta distribution given by  $E(p)$  can be defined as a local reputation value  $C_{ab}$ .

$$C_{ab} = E(beta(p|u+1, v+1)) = \frac{u+1}{u+v+2} \tag{2}$$

The feedback data structure resulting from this process can be apparently represented as a weighted directed multigraph, that we call trust network in the sequel. Its node set coincides with the set of agents and the set of edges with the set of interactions between them. The content consumer is assumed to be the source of the edge corresponding to any given interaction. The weight assigned to an edge represents the content consumer’s feedback on the content provider’s trustworthiness in the corresponding interaction. Such a feedback set might be used if one wants to deal differently with recommendations of other agents and direct content provisions.

A given agent in order to use the feedback that is available in the trust network to evaluate the trustworthiness of any other agent, it have to assess the trustworthiness of a node (say, agent  $j$  from Fig.2), compute the weighted average of the experiences of the nodes which interacted with that node (agent  $u$  and  $v$ ) where the weights are the trustworthiness values of the feedback originators. This *global reputation value* can be expressed by the formula:

$$t_j = \sum_{e \in in(j)} w_e \cdot s_e \cdot \frac{t_{src(e)}}{\sum_{f \in in(j)} t_{src(f)}} \tag{3}$$

where  $in(j)$  is the set of all edges ending at node  $j$ ,  $w_e$  is the feedback belonging to the edge  $e$  which is equal to  $C_{ab}$ ,  $s_e$  is the semantic similarity belonging to  $e$  which defied below and  $t_{src(e)}$  is the trustworthiness of the originator of this feedback.

All this reasoning leads us to define a reputation multi-agent system based on [3].

**Definition 1** (Reputation multi-agent system). A reputation multi-agent system can be defined as  $(G, W, OP, T, O)$ , where  $G$  is a directed weighted graph  $(A, V)$ , with  $A$  being the set of agents and  $V$  the set of edges which are assigned weights drawn from the set  $W$ .  $OP$  is an algorithm that operates on the graph and outputs a specific value  $t \in T$  for any agent given as its input.  $O$  is the set of ontology based on agents.

The problem of the trust management based on the agents’ reputations can now be stated simply as follows: define a strategy based on  $O$  to aggregate the available feedback (algorithm  $OP$ ) and output an estimate of the trustworthiness of any given agent (set  $T$ ) so that trustworthy behavior of the agents is encouraged. Different from the other reputation model, we use semantic similarity between agents to aggregate the feedback.

### 3.3 Aggregate Based on Semantic Similarity

Since trust is defined in the context of similarity conditions, the more similar the two agents are the great their established trust would be considered [13]. So, we can utilize semantic information of agents compute the agent’s trust value. In more detail, in order to aggregates the feedback according Equation 3, we use semantic similarity between agents to compute  $s_e$ . Here, we mainly focus on the global

shared homogeneous ontology. Much effort in the research on how to mapping and emerging heterogeneous ontology has been made in recent years [5].

A single document (or the content of document), for example, can be classified into at least one topic. So, the semantic reputation model measures the similarity between agents in the same way. We use  $P = \{ \langle T_i, \lambda_i \rangle, i=1, 2, \dots, m \}$  to describe the participant topic, where  $T_i$  denotes an agent's topic, and  $\lambda_i$  shows the degree of interest to  $T_i$ . The similarity between two different agents is described as the similarity among the sets of topics.

$$Sim(P^1, P^2) = Sim(\{ \langle T_i, \lambda_i \rangle, i=1, 2, \dots, m \}, \{ \langle T_j, \lambda_j \rangle, j=1, 2, \dots, n \}) \quad (4)$$

In case the agents in the network share a common topic hierarchy our aggregate algorithm exploits the semantic similarity between agents. The study of semantic similarity between lexically expressed concepts has been a part of natural language processing for many years. A number of measurement methods have been developed in the previous decade, among which [12] gives the best results.

$$Sim(T_1, T_2) = f_1(l) \cdot f_2(h) = \begin{cases} e^{\alpha l} \cdot \frac{e^{\beta h} + e^{-\beta h}}{e^{\beta h} - e^{-\beta h}} & \text{if } (T_1 \neq T_2) \\ 1 & \text{if } (T_1 = T_2) \end{cases} \quad (5)$$

Here,  $l$  counts the shortest path length between  $T_1$  and  $T_2$  and  $h$  counts the hierarchy depth from the leave of  $T_1$  and  $T_2$  to the top of the concept.  $\alpha > 0$  and  $\beta > 0$  are parameters scaling the contribution of shortest path length and depth, respectively. Based on the method, the following equation is proposed to measure the similarity between two agents. [14]

$$Sim(P_1, P_2) = \sum_{j=1}^{|P_2|} \sum_{i=1}^{|P_1|} [Sim(T_i, T_j) \times (\lambda_i \times \lambda_j)] \quad (6)$$

Here,  $|P_1|$  and  $|P_2|$  are the topic numbers in the two agents. The similarity between the sets of topics of each other is calculated by summing up products of the similarity value between two topics separately selected from  $P_1, P_2$ .

## 4 Simulation Results and Performance Evaluation

We evaluate our approach in a simulation of a peer-to-peer network based multi-agent system from original Gnutella-like network. For the sake of simplicity, each node in our system plays only one role at a time, either the role of content provider or the role of an agent. Every agent only knows other agent s directly connected with it and a few content providers at the beginning.

Every agent has a topic vector. The topic is composed of five elements: music, movie, image, document and software. The value of each element indicates the strength of the agent's topic in the corresponding content type. The content the agent wants to share are generated based on its topic vector. Every agent keeps two lists. One is the agent list that records all the other agents that the agent has interacted with and its trust values in these agents. The other is the content provider list that records

the known content providers and the corresponding reputation data representing the agent’s trusts in these content providers. Each content provider has a capability vector showing its capabilities in different aspects, i.e. providing content with different types, qualities and download speeds. Our experiments involve 10 different content providers and 400 agents. Each agent will gossip with other agents periodically to exchange their reputation data. According to [12],  $\alpha$  and  $\beta$  can be set 0.2 and 0.6 separately.

(1) Trust-based mechanism vs. Normal mechanism

The goal of this experiment is to see if a SemTrust-based trust model helps agents to select content providers that match better their preferences. Therefore we compare the performance (in terms of percentage of successful recommendations) of a system consisting of agents with SemTrust-based trust model and a system consisting of agents (without SemTrust) that represent general trust, not differentiated to different aspects. Successful recommendations are those positive recommendations when agents are satisfied with interactions with content providers with good reputation. If a agent gets a negative recommendation for a content provider, it will not interact with the content provider. Figure 3 shows that the system using SemTrust-based reputation performs better than the system without reputation mechanism, especially when the number of interactions is large. This is profitable for the large, uncensored Semantic Web. In some sense, an agent’s trust networks based on SemTrust can be viewed as the model of a specified content provider from the agent’s personal perspective. In the real web semantic, the model of content providers might be more complex and required the use of a more complex semantic-based mechanism.

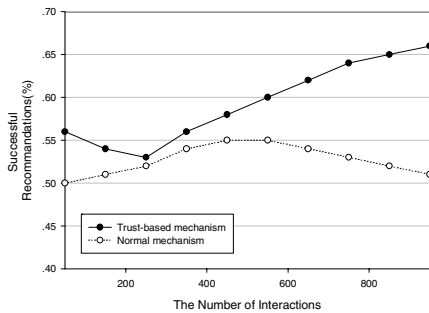


Fig. 3. SemTrust vs. normal mechanism

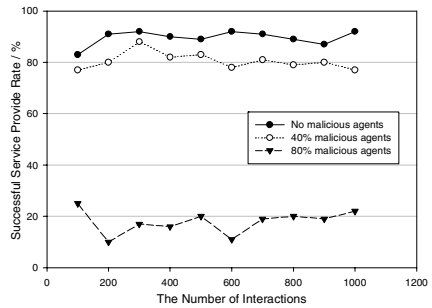


Fig. 4. Successful service provide rate with different malicious agent number

(2) Successful Service Provide Rate with malicious agents

The main goal of this experiment is to examine the result of successful service provide rate service when existing malicious agents. In short, malicious agents operating under threat model. A malicious agent try to assign positive trust ratings to any other malicious agent they get to interact with while participating in the network. Figure 4 shows that even when the malicious agents ratio is about 40%, the successful service provide rate is still at a high level, neatly 80%. But when the malicious agents ratio is

about 80%, the successful rate decreases dramatically. This is because the number of malicious agents is so large, and affects agents' behavior deeply.

(3) Successful execution ration

These experiments examine the successful execution ratio in the multi-agent system in the condition of different number of tasks and agents. Tasks in semantic web such as searching and queering is important to users, the successful execution of tasks is valuable. We set 30% of agents in the system with 80% ratio of failing to fulfill the task. Figure 5 shows that the trust-based mechanism is helpful when executing tasks because agents are more likely to outsource the task to trusted agents which can fulfill the work more successfully. In figure 6, with the number of agents increasing, the successful execution ratio also increases and performs better than normal mechanism. This shows that SemTrust is suit for large scale multi-agent system in semantic web, which is promising to use it in the real world.

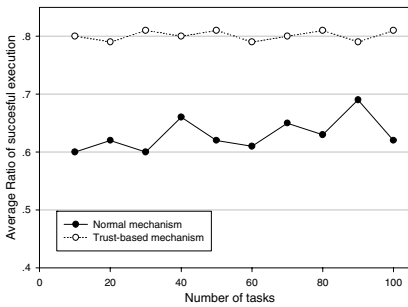


Fig. 5. Average ratio of successful execution with different tasks

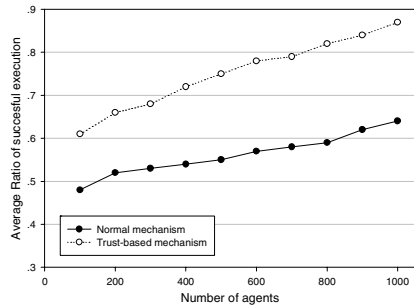


Fig. 6. Average ratio of successful execution with different agents

## 5 Conclusions

In this paper, we propose a semantic-based reputation system to solve the problem of lack trust in Semantic Web. We evaluated our approach in a simulation of a content sharing system in a multi-agent based Semantic Web. Our experiments show that the system with SemTrust where users communicate their experiences (reputation) outperform the system where users do not communicate with each other, and our approach is more robust with security under condition where existing malicious agents. SemTrust can also helpful agents fulfill the task more successfully by outsourcing the work to trusted agents.

## Acknowledgements

This research was partially supported by the National Natural Science Foundation of China under grant of 60173026, the Ministry of Education key project under grant of 105071 and SEC E-Institute: Shanghai High Institutions Grid under grant of 200301.

## References

1. Bonifacio, M., Bouquet, P., Traverso, P.: Enabling Distributed Knowledge Management: Managerial and Technological Implications. *Novatica and Informatik/ Informatique*, III(1), (2002)
2. Resnick, P., Zeckhauser, R., Friedman, R., et al.: Reputation systems, *Communications of the ACM* 43 (12) (2000) 45–48
3. Despotovic, Z., Aberer, K.: P2P reputation management: Probabilistic estimation vs. social networks. *Computer Networks* 50 (2006) 485–500
4. Loser, A., Tempich, C., Quilitz, B., et al.: Searching dynamic communities with personal indexes. in 3rd. International Semantic Web Conference (ISWC) Galway, (2005)
5. Jin, H., Wu, H., et al.: An Approach for Service Discovery based on Semantic Peer-to-Peer. *Proceedings of Tenth Asian Computing Science Conference Data management on the Web (ASIAN'05)*, Kunming, China, Dec. 7-9, (2005)
6. Arumugam, M., Sheth, A., Arpinar, I. B.: Towards Peer-to-Peer Semantic Web: A Distributed Environment for Sharing Semantic Knowledge on the Web. Technical report, Large Scale Distributed Information Systems Lab, University of Georgia, (2001)
7. Haase, P., Broekstra, J., Ehrig, M., et al.: Bibster-a semantics-based bibliographic peer-to-peer system. in *Proceedings of the International Semantic Web Conference (ISWC)*, (2004)
8. Ernst, J.: Peer-to-Peer infrastructure supporting the semantic web, *Int. Semantic Web Symposium*, Stanford Univ. (2001)
9. Jiang, Y.C., Xia, Z.Y., et al.: Autonomous trust construction in multi-agent systems-a graph theory methodology. *Advances in Engineering Software* 36 (2005) 59–66
10. Gupta, M., Judge, P., Ammar, M.: A reputation system for peer-to-peer networks. In *Proceedings of the NOSSDAV'03 Conference*. Monterey, CA, (2003)
11. Xiong, L., Liu, L.: Building trust in decentralized peer-to-peer communities. In *Proceedings of the International Conference on Electronic Commerce Research*, (2002)
12. Li, Y., Bandar, Z., McLean, D.: An Approach for measuring semantic similarity between words using semantic multiple information sources. In *IEEE Transactions on Knowledge and Data Engineering*, vol. 15, (2003)
13. Ziegler, C.N., Lausen, G.: Analyzing Correlation between Trust and User Similarity in Online Communities. *Proc. of the 2nd International Conference on Trust Management*, (2004)
14. Chen, H.H., Jin, H., Ning, X.M., Semantic Peer-to-Peer Overlay for Efficient Content locating”, *Proceedings of MEGA'06*, Harbin, China, Jan.16-18, (2006)
15. Ramchurn, S., Sierra, C., Godo, L., Jennings, N.: Devising a trust model for multiagent interactions using confidence and reputation. *International Journal of Applied Artificial Intelligence*, 18(9–10) (2005) 91–204
16. Papoulis A., Pillai, S. *Probability, Random Variables and Stochastic Processes*. Simplified Chinese translation edition, McGraw-Hill, 4th ed., 80~81, (2004)

# Ontological Modeling of Virtual Organization Agents\*

Liao Lejian, Zhu Liehuang, and Qiu Jing

School of Computer Sciences and Engineering  
Beijing Institute of Technology, Postcode 100081, Beijing, China  
liaolj@bit.edu.cn, liehuangz@bit.edu.cn, qiujing@bit.edu.cn

**Abstract.** Cross-organizational interoperability and coordination are major challenges to Virtual Organization(VO) applications. Multi-agent systems combined with Semantic Web are promising approach for solving the challenging problems. In this paper, a semantic Web enabled multi-agent platform for logistic VO supporting is presented. The issue of extending OWL with multi-attribute constraints for VO modeling is addressed. A constraint rule language SWOCRL is proposed which is based on OWL and SWRL with constraint extension and class-scoped restriction. Important VO concepts such as organizations, activities, resources, contracts, interactions and their logistic specializations are described with OWL plus SWOCRL.

**Keywords:** semantic Web, agent, virtual organization modeling, constraint rule, ontology.

## 1 Introduction

Virtual organization is a conceptual model for Internet-based cooperation across organizations. A VO system consists of a collection of autonomous entities (either organizations or individuals). Each entity has certain capabilities or resources that are published to the Internet. Multiple entities may dynamically form a contracted coalition to pursue their respective objectives. All the entities in a virtual Organization system share a virtual space in which they cooperate and compete with each other. A Web-based computer platform that supports VO faces a number of technical challenges, including how an organizational agent finds suitable business partners from Internet; how heterogeneous agents interact with each other and coordinate their activities; how an organizational agent plans its activities to achieve its maximal goals; and how agents negotiate to achieve agreements in both coalition formation and cooperative execution. Semantic Web (SW) combined with multi-agent systems is a promising technology for solving these challenging problems. SW provides well-defined and standardized framework for ontology-based semantic interoperability, dynamic service integration, and dynamic contracting of business agreements between organizational agents. For SW to solve these problems, it is necessary for SW-enabled VO agents to understand common conceptualization across business domains. VO conceptualization extensively involves modeling of such concepts as organizations, time and space, processes and activities, physical resources, interaction

---

\* This work is funded by National Science Foundation of China under grant No. 60373057.



and negotiation, policies and agreements, as well as their relations. A typical feature of such conceptualization is the representation of constraint-relations between the attributes of the concepts, especially quantitative relations between the quantities. Without the presentation of these constraints, the inherently attribute-constrained concepts would miss the most significant meaning. This is apparent for the modeling of concepts as time, space, and commerce. Furthermore, such constraints constitute the basic knowledge for problem-solving supporting VO applications.

In SW pyramid, information modeling mainly involves RDF(S) layer, ontology layer, and logic layer. The modeling of general multi-attribute constraints should be above ontology layer because it is not supported in current ontology layer language OWL[1] which is based on some form of description logic(DL). Syntactically SW rule language RuleML[2] should cover any datalog rules including constraint expressions. But semantically constraint expressions are interpreted according to built-in domain theories which are beyond the expressibility of datalog rules. In addition, a RuleML rule that spans over the instance stores of multiple classes exhibits overly combinatorial complexity that is unnecessarily wasteful for the modeling of multi-attribute constraints within a class. In this paper, we propose a constraint language which is based on OWL and SWRL[3], with constraint extensions, and demonstrate its applicability in VO modeling with our urgent logistic scenario.

The organization of the paper is as follows. Section 2 describes the background of the work and a fictive scenario. Some related technology are reviewed and analyzed in section 3. In section 4 we sketch our design of a semantic Web enabled multi-agent platform for logistic VO supporting. A semantic Web constraint language SWOCRL is defined in section 5, and some examples of concepts about VO and logistics with this language combined with OWL are presented in section 6. Section 7 gives a concluding remark to the work.

## 2 Background and a Scenario

The work is part of our Urgent Logistics project which investigates the technology of Web-agent based intelligent Logistic information platform for the urgent nation-wide mobilization of required resources on large-scale disasters. The following is a scenario for our investigation. Suppose that a large-area devastating disaster, like the Pacific tsunami occurred in Asia in 2004 or Hurricane Katrina swept over New Orleans in 2005, erupted in some remote area of China. A committee is set up by the State Council to organize and command all the urgent rescuing, recovery, and aiding activities in the disaster area. Subordinate to the committee there are several functional groups responsible for specific tasks. Group G-RedCross is responsible for the reception and registration of all donations of money and goods by individuals and organizations from the society. Group G-finance is responsible for the management of all the money from the social donation and government finance. Group G-Logistics is responsible for the supply of the necessity goods and equipments to the disaster area, including the purchase, storage and shipment of the goods and equipments. In addition to the internal workflow activities between the groups within the committee, there are coordination activities around the disaster-rescuing between the committee groups and organizations in the society as follows.

- 1) G-RedCross publishes to the society all the donations that have been received so far, and the demanding resources by the disaster people, with resource types and quantities.
- 2) G-finance publishes all the financial accounts for public supervision.
- 3) G-Logistics searches for (a) merchants and manufacturers that sell or provide the demanded goods or equipments; and (b) logistic companies that perform shipment of resources from the stores of the merchants and manufacturers to the disaster area. Then G-Logistics selects a certain number of best candidates, in terms of product quantities, prices, reputations, services, and etc., for further negotiation.
- 4) After 3), G-Logistics interacts and negotiates with the selected candidates for agreements about the sales of the required products or services.
- 5) After contracting agreements with some merchants, manufacturers or logistic companies, G-Logistics would enforce the agreements by checking the partners' progresses in due time, checking and accepting the goods supplied by the merchants or manufacturers, handing over the goods to the logistic company, and finally accepting the delivery-reception acknowledgement from the local government.
- 6) In addition to the agencies of the rescuing committee, the coordination process involves other entities in the supply-chain networks.
- 7) Donating organizations or individuals either donate money to G-RedCross, or donate goods or equipment and then contact with G-Logistics with shipment issues.
- 8) Merchants and Manufacturers publish their products goods and merchants, and interact and negotiate with customers like G-Logistics for sales. They also have business transactions with their goods providers.
- 9) Logistic companies publish their business and negotiate with interested customers such as G-Logistics for shipment tasks. To perform a shipment task, a logistic company may negotiate with warehouses for intermediate storage in the process of transportation, and with the gas-stations along the highway the transportation pass along for intermediate refueling.
- 10) Warehouses provide relayed storage for the goods during the transportation process. They periodically publish their spare spaces for goods storage.
- 11) Gas-stations periodically publish their provision quota in a day, a week, or a month, with policies for overly large mount of consumption request.

More involving organizations in the scenario could be listed if we further trace the relevant entities along the supply-chain networks. To realize the scenario in Internet with VO conception, it would be natural to have all these organizations modeled as their computational counterparts, i.e., autonomous organizational agents that act on their behalf. Thus the organizational coordination could be maximally automated through an agent-based VO supporting platform. Further more, such VO supporting platform should include facilities to support basic business activities with Internet, such as service discovery and integration across Web, and coordination between organizations. Service discovery/integration and business coordination in general have attracted widespread attentions. The main concerns of this paper focus on some issues about commerce-related semantic Web modeling, which are of less attention but, as we believe, no less importance from the viewpoint of VO applications.

### 3 Related Technology

#### 3.1 Semantic Web: Interoperability vs. Conceptual Modeling

As a blueprint of future Web, a main objective of SW is to achieve maximal interoperability through standardized semantic annotation of Web information. The core of semantic Web in the prevailing layered architecture of semantic Web is RDF(S) layer and ontology layer. Some pitfalls of the current layered standards, especially between RDF(S) and OWL, have been analyzed[4]. One pitfall is the interoperability problem between RDF(S) and OWL. Another is the inadequacy of OWL for conceptual modeling. For instance, OWL is unable to express constraints between object roles such as “A person’s age is less than those of his parents”, or “the difference between the end time and start time of an activity equal to the its duration”. The inadequacy stems from the fact that OWL was designed for interoperability and there is a gap between representational requirements for interoperability and for conceptual modeling respectively. Specific to VO and electronic commerce, the conceptual modeling commonly involves complex constraint relations between the attributes of domain objects, which are beyond the expressibility of OWL. Efforts towards the problem included extension of classical description logic with data-type predicates[5]. In [6], the importance of expressive constraints for SW was realized and a constraint language was defined with SW ontology, but it had no inheritance to SW languages and cannot be taken as an constraint-enhancement to DL-based concept language.

#### 3.2 VO Modeling

In [7], a VO modeling framework was proposed which was based on constraint logic. The modeling framework makes full advantages of the representational and problem-solving powers of constraint logic programming, but with little concern for the interoperability and coordination between different organizations, as well as their agents. The business-related concepts such as business activities and entities should be modeled in ontology as a part of consensus between the organizational agents. The common concepts in general VO include time, events, activities, interactions, services, resources and organizations. Although such concepts have been modeled in various applications, most of them are modeled in ad hoc that is specific to the applications. PSL[8] is one of few works that construct enterprise ontology, including time, events, activities, and resources, in a systematic fashion. But as a pioneering work on enterprise ontology construction, PSL is written in KIF rules rather than description logics. Although the rules give more refined characterization, they are not interoperable with current semantic Web formulation at conceptual level since the conceptual structure of the ontology is not explicit. For example, they cannot be feasibly reused by the existing semantic Web service matchmaking algorithm[9].

### 4 A Service-Oriented Multi-agent Architecture for Logistic VO

With the urgent logistics background, we design a service-oriented agent-based VO supporting platform. The architecture consists of a set of service agents and task

agents. A task agent, such as the organizational agent for G-Logistics in the scenario, is based on BDI model which has beliefs, goals, actions, and strategies in its body. It plans its actions for the goals and re-plans to adapt dynamic environment events. A service agent, such as those for merchants, logistic company, gas-stations in the scenario, publish their service as semantic Web service descriptions with OWL-S and behavior pattern as constraints. Some service agents may at the same time be task agents in that they act in BDI mode and as clients to other service agents during service execution. As a kernel component of the VO supporting platform, an OWL-S enhanced semantic UDDI database stores all the service items for service discovery. Service discovery is invoked with a semantic service request issued by a task agent to a service matchmaker that matches the request against service items in the UDDI database. In our VO supporting platform, several features are added to the original semantic Web service matchmaking algorithms. The first feature is the representation and exploitation of geographical and organizational information in the service matchmaking; the second feature is the consideration of security and trust qualification of the requesters, i.e., the answers to a request will not include those services that require trust qualification higher than what the requester has. The third feature is the requests for quantified resources which are common in commerce.

Other components of the VO supporting platform include Web-service wrapped agent communication infrastructure, semantic Web and ontology management, editing and reasoning modules, general VO ontology, agent registration and management modules, knowledge base tools (currently f-Logic and Fuzzy Clips), semantic-web service enabled model base for authorized agents, GIS sever, security management modules, and etc.

## 5 SWOCRL: A Language for SW Modeling with Object Constraints

A generic motivation of this work is “engineering the semantic Web” towards electronic commerce applications. A starting point is to extend rigid description-logic formulas of OWL with more flexible object constraints that are common in VO modeling. We propose a constraint languages SWOCRL (Semantic Web Object Constraint Rule Language) which is based on OWL and SWRL. It relies on the conceptual structure defined by OWL assertions, and uses rules to infer object structures and to impose constraints on object attributes.

### 5.1 SWOCRL Syntax: Rules for Object Constraints

SWOCRL modifies SWRL with some extensions and specializations. The abstract syntax of SWRL is mainly as follows[3]:

```
rule ::= 'Implies(' { annotation } antecedent consequent ')'
antecedent ::= 'Antecedent(' { atom } ')'
consequent ::= 'Consequent(' { atom } ')'
atom ::= description '(' i-object ')' | individualvaluedPropertyID '(' i-object i-object ')' |
        datavaluedPropertyID '(' i-object d-object ')' | 'sameAs (' i-object i-object ')' |
        'differentFrom (' i-object i-object ')'
```

*i-object* ::= *i-variable* | *individualID*  
*d-object* ::= *d-variable* | *dataLiteral*  
*i-variable* ::= '*I-variable*(' *URIreference* ')'  
*d-variable* ::= '*D-variable*(' *URIreference* ')'

SWOCRL extends SWRL by allowing

1) Some attributes (unique roles) and attribute paths as constraint variables. An RDF statement *rdfs:subPropertyOf*(*swocrl:constraintAttribute*, *owl:UniqueProperty*) defines a special class *swocrl:constraintAttribute* of such constraint attributes.

2) Multi-attribute constraints as atoms in both antecedents and consequents of rules. Variables introduced in antecedents are taken to be a universal variable; Variables introduced in consequents are taken to be existential ones.

SWOCRL then adds the following syntax rules to SWRL:

*atom* ::= *constraint*  
*constraint* ::= *predicate-name* '(' {*d-object*} ')'  
*d-object* ::= *d-path-exp*  
*d-path-exp* ::= *d-attribute* | *d-attribute* '.' *d-path-exp*  
*i-object* ::= *i-path-exp*  
*i-path-exp* ::= *i-attribute* | *i-attribute* '.' *i-path-exp*

SWOCRL specializes SWRL such that a SWOCRL rule only specifies assertions of just one class, featuring it as an object constraint language. Such specialization is desirable to circumscribe the complexity of rule reasoning. The specialization includes the following restrictions to the above syntax:

- 1) The first atom in the antecedent must be fixed as *class-name* '(' *I-variable* ')', which indicates the class that the rule asserts about.
- 2) For the following atoms, a unary class description atom must have an instantiated argument, i.e., either a constant individual, or an *I-variable* that occurs in a preceding atom; A binary property atom must have the first argument instantiated.

## 5.2 Operational Semantics Based on Constraint Handling

The operational semantics of SWOCRL is a forward inference process with constraint store accumulation in sense of concurrent constraint programming[10]. A SWOCRL rule is typically fired when an object is constructed or new information is added to it. The antecedent is checked against the object and the consequent is enforced if the check succeeds. The inference engine incrementally fills the values of object roles, constructs and accumulates constraints on object attributes, refines their ranges and derives their values. Note that a constraint in an antecedent performs an asking test while a constraint in the consequent performs a telling imposition in sense of CCP terms. If the information accumulation process finally arrives at quiescence while there still remains unresolved constraint, a domain-specific constraint solver will be called to check the consistency of the constraint store.

## 6 VO Ontological Modeling

The ontology involved in the logistic applications includes organizations, activities, interactions, resources, geography, transportation, and etc. Here we show some DL-form concept descriptions and associated SWOCRL constraint rules that are typical in VO modeling in general and urgent logistics in particular. Simplifications are made for clarity and space limitation reasons.

### 6.1 Organizational Modeling

We view an organization as a service actor with certain composition structure and functioning for some services. The following lists DL axioms for organization, logistic-organization and warehouse as well as their functional services, starting from the concept of *Service-actor*.

*Service-actor*  $\sqsubseteq \forall has-service. Actor-service$   
*Actor-service*  $\sqsubseteq \forall has-action. Action \cap \forall has-goal. Goal$   
*Organization*  $\sqsubseteq Service-actor \cap \forall has-service. Organization-service \cap$   
 $\forall has-subordinate. Organization \cap \forall subordinate-to. Organization \cap$   
 $\forall has-position. Organizational-position \cap \forall located-in. Location$   
 Note that *subordinate-to* is the inverse property of *has-subordinate*.  
*Organization-service*  $\sqsubseteq Actor-service \cap \forall has-action. Organization-action \cap$   
 $\forall has-goal. Organization-goal$

The following is logistic specialization of general organization conception.

*Logistic-organization*  $\sqsubseteq Organization \cap \forall has-service. Logistic-service$   
*Logistic-service*  $\sqsubseteq Organization-service$   
*Warehouse*  $\sqsubseteq Logistic-organization \cap \forall has-service. Warehouse-service$   
*Warehouse-service*  $\sqsubseteq Logistic-service \cap \forall stored-goods. Goods \cap$   
 $=1 current-amount. Integer \cap \forall recent-storage-plan. Warehouse-plan \cap$   
 $=1 has-storage-capacity. Integer$   
*Warehouse-plan*  $\sqsubseteq =1 time-period Time \cap =1 input-amount. Integer \cap$   
 $=1 output-amount. Integer$

Roles *input-amount* and *output-amount* are defined as constraint attributes:  
*input-amount*: *ConstraintAttribute*, *output-amount*: *ConstraintAttribute*,

A capacity constraint for the storage in a time is imposed on a warehouse, i.e. Current amount + input - output < storage capacity. Such constraint is expressed in the following SWORCL rule with built-in constraints *sum*(*x*, *y*, *z*) (as  $x+y=z$ ) and *greaterThan*(*x*, *y*) ( as  $x > y$ ). For a compact writing, we denote variables as capitalized-letter prefixed names instead of their URI form in above grammar.

*Implies* (  
*Antecedent* (*Warehouse-service*(*X-ws*) *has-storage-capacity*(*X-ws* *X-sc*)  
*has-storage-plan*(*X-ws* *X-sp*) *current-amount*(*X-ws* *X-ca*))  
*Consequent* (*sum*(*X-ca* *X-sp*.*input-amount* *Z1*) *sum*(*X-sp*.*output-amount* *X-sc* *Z2*)  
*greaterThan*(*Z1* *Z2*)  
 ))

## 6.2 Activity Modeling

In our work, activities are modeled according to their goals, participants, compositional and temporal relation.

$$\text{Activity} \subseteq \forall \text{has-service. Actor-service} \cap \forall \text{has-role. Actor} \cap \\ \forall \text{has-subactivity. Activity} \cap \forall \text{succeed-to. Activity}$$

Activities can further have time attached for refined temporal description.

$$\text{Timed-activity} \subseteq \text{Activity} \cap \forall \text{in-period. Time-Interval}$$

$$\text{Time-Interval} \subseteq =1 \text{start-time. Time-point} \cap =1 \text{end-time. Time-point} \cap \\ =1 \text{duration. Time-Duration}$$

*start-time:ConstraintAttribute, end-time:ConstraintAttribute,*  
*duration:ConstraintAttribute*

The relation of the start time, end time, and duration of a time interval can be represented as SWORCL rule:

*Implies (*  
*Antecedent (Time-interval (X-ti)*  
*Consequent (time-sum(X-ti.start-time X-ti.duration X-ti.end-time)))*  
*time-sum* is the temporal counterpart of *sum*. Specific to the logistic field, we have activities such as movements and transportation  
*Movement*  $\subseteq$  *Timed-activity*  $\cap$   $\forall$  *along-path.Path*  $\cap$   $=1$  *average-velocity.Velocity*  
*Path*  $\subseteq$  *Spatial-entity*  $\cap$   $\forall$  *on-path.Location*  $\cap$   $=1$  *start-location.Location*  $\cap$   
 $=1$  *end-location.Location*  $\cap$   $=1$  *length.Length-metric*  
*start-location* and *end-location* are sub-properties of *on-path*:  
*start-location*  $\subseteq$  *on-path*, *end-location*  $\subseteq$  *on-path*

The following SWORCL rules express temporal-spatial assertions for *Movement* that the actor is at the starting location at the start, and at the end location in the end.

*Implies (*  
*Antecedent ( Movement(X-mv) has-role(X-mv X-rl) along-path(X-mv X-ap)*  
*start-location(X-ap X-sl) end-location(X-ap X-el) in-period (X-mv X-tp))*  
*Consequent(at-location(X-rl X-sl X-tp.start-time)*  
*at-location(X-rl X-el X-tp.end-time)))*

For *Movement* there would be another rule to specify the relation between the length, time and velocity, but omitted here. The following are axioms about logistic transportation.

$$\text{Transportation} \subseteq \text{Movement} \cap \forall \text{has-actor. Transportation-actor} \cap \\ \forall \text{with-traffic. Traffic-System} \cap \forall \text{has-load. Transportation-load}$$

$$\text{Traffic-system} \subseteq \text{Facility} \cap \forall \text{with-traffic-vehicle. Vehicle} \cap \forall \text{in-traffic-line. Traffic-line}$$

$$\text{Highway} \subseteq \text{Traffic-system} \cap \forall \text{with-traffic-vehicle. Automobile} \cap \\ \forall \text{in-traffic-line. Road-line}$$

For *Transportation*, there would be a rule to denote the constraints that *Transportation-load* will be moved from *start-location* to *end-location* during the

*time-interval*, and that the *Path* of the *Movement* must be consistent with the *Road-line* when the *Transportation* is on the *High-way*.

### 6.3 Resource Modeling

In VO trading activities, the concept of *resources* is in widespread uses. By resources we mean objects in contexts of activities that use them.

#### 6.3.1 Quantified Resources

An especially important concept in VO is a quantity volume of resources in which the quantity rather than the individual elements of a set of entities is concerned, such as 20 trucks or 2 millions of quilts. In syntactic sense, such a concept of quantified volume involves two components: a non-negative number  $n$  denoting the quantity of the resources and a class  $c$  denoting the type of the resources. In semantic sense, the extension of such a concept is an  $n$ -element subset of class  $c$ . A straightforward representation of quantified resources based on such intuition would be a second-order class. But introduction of higher-order classes would undesirably complicate the conceptual reasoning. To bypass such difficulty, we do not take a quantified resource directly as a second-order instance. But instead, we view it as a first-order individual that affiliates the members of the collection through a multi-value property, named *has-element*. With this property, we can uniformly denoting resources either as individuals or as collections.

$$Resource \sqsubseteq \forall has\text{-}element. Element\text{-}Class \cap \forall has\text{-}context Resource\text{-}Context$$

Where *Element-Class* is supposed to be the top class of resource elements of concern; *has-context* specifies the context features related to the activity using the resource. The features are modeled as subclasses of *Resource-Context* which contains attributes on which such features depend, as described in below. Since we are frequently concerned with the total number of the collections, we define a subclass *Quantified-resource* of *Resource* that has a property *total-number*.

$$Quantified\text{-}resource \sqsubseteq Resource \cap =1\ total\text{-}number. Integer$$

#### 6.3.2 Resource Contexts

In addition to its composition, a resource may exhibit external features that are associated with the activities using it, such as features related allocation. Several main features of this sort are *allocability*, *sharability*, *reusability*, and *dividability*. *Allocability* indicates if a resource can be allocated independently to an activity. For example, a classroom can be allocated to a class, but a number of classroom seats cannot. *Sharability* indicates if a resource can be allocated to more than one activities in the same time. A highway is sharable to many vehicles with respect to transportation activities, while a classroom can only be used by one class at any time. *Reusability* indicates if a resource can be reused to other activities after its use by one activity. A classroom is reusable w.r.t to class while a gallon of fuel will consume away after it is used out. *Dividability* indicates if a resource can have its parts allocated to other activities. A gallon of fuel can be divided and allocated to two



vehicles for running, while a vehicle cannot be divided into two pieces while still perform transportation respectively. These features are modeled as subclasses of

*Resource-Context*.

$Resource-Context \sqsubseteq =1with-activity.Activity$

$Resource-Context \equiv Allocable-R-Ctx \cup Non-allocable-R-Ctx$

$Allocable-R-Ctx \cap Non-allocable-R-Ctx = \Phi$

$Resource-Context \equiv Sharable-R-Ctx \cup Non-sharable-R-Ctx$

$Sharable-R-Ctx \cap Non-sharable-R-Ctx = \Phi$

$Resource-Context \equiv Reusable-R-Ctx \cup Non-reusable-R-Ctx$

$Reusable-R-Ctx \cap Non-reusable-R-Ctx = \Phi$

$Resource-Context \equiv Dividable-R-Ctx \cup Non-dividable-R-Ctx$

$Dividable-R-Ctx \cap Non-dividable-R-Ctx = \Phi$

The definition of resource contexts as a multi-value property indicates the fact that the activity-related resource features are not uniform for one resource type. For different activities a resource may exhibit opposite features. For example, a bus is sharable to different passengers but not sharable between different running routes. This can be represented as follows:

$Bus-Resource \sqsubseteq Resource \cap =1has-element.Bus$

$Bus-resource \sqsubseteq \forall has-context. (\neg \forall with-activity.Passenger-Riding \cup Sharable-R-Ctx)$

$Bus-resource \sqsubseteq \forall has-context. (\neg \forall with-activity.Route-allocation \cup Non-sharable-R-Ctx)$

Quite often the identification of the features may depend on more attributes such as time. In such cases the class *Resource-Context* can be extended to subclasses with the required attributes. For instance, an extension of *Resource-Context* with time can be defined as follows:

$Temporal-Resource-Context \sqsubseteq Resource-Context \cap =1 in-time.Time$

## 6.4 Contract Modeling

Contracts are important objects in VO. Intuitively we consider a contract as a set of statements, especially commitments of obligations and rights, made by a set of contractors over one or more issues. We especially assume the existence of those issues about the validation of a contract, such as the validation time and conditions.

$Contract \sqsubseteq =1has-contract-No.String \cap =1has-contract-title.String \cap$

$\geq 2 has-contractor.Legal-actor \cap =1signing-time.Time \cap$

$\forall has-contract-issue.Contract-Issue \cap \exists has-contract-issue.Contract-Valid-Issue$

We distinguish contracts about services between the service provider and service requester, who are both the contractors of the contract.

$Service-contract \sqsubseteq Contract \cap =1has-service-provider \cap =1has-service-requester \cap$

$\forall has-contract-issue.Service-issue$

$Service-issue \sqsubseteq Contract-issue \cap \forall has-service.Service$

The fact that both service provider and service requester are the contractors of the contract is expressed as role subsumption axioms:

$$\text{has-service-provider} \sqsubseteq \text{has-contractor}, \text{has-service-requester} \sqsubseteq \text{has-contractor}$$

As an illustration, consider a special class of service contracts, commodity reservation contracts.

$$\text{Commodity-reservation-contract} \sqsubseteq \text{Service-contract} \cap \forall \text{has-contract-issue} \text{Commodity-reservation-issue}$$

$$\begin{aligned} \text{Commodity-reservation-issue} \sqsubseteq & \text{Service-issue} \cap \forall \text{provided-commodity} \text{Commodity} \cap \\ & \forall \text{has-service} \text{Commodity-reservation} \cap =1 \text{has-quantity} \text{Integer} \cap \\ & =1 \text{has-price} \text{Commodity-price} \cap \forall \text{has-delivery} \text{Delivery} \cap \\ & \forall \text{has-punishment} \text{Punishment} \end{aligned}$$

$$\text{Commodity-price} \sqsubseteq =1 \text{per-unit} \text{Unit} \cap =1 \text{price-value} \text{Integer}$$

## 6.5 Interaction Modeling

Currently simple forms of interaction such as request-response and one-way sending are established as stereotyped patterns in popular as web-service based enterprise process execution languages. In reality, however, the requirements for flexible interaction are widespread in the areas of grid services, VO and trust computing. For such interactions to proceed, it is firstly necessary for the servers to openly publish the rules of interactions as a part of the service descriptions. Secondly, the descriptions of the interaction rules should be understandable not only to the designers of the client programs, but also to the programs themselves so that the interactions are amenable to dynamic reasoning for the extraction of their messaging properties. Such reasoning is important in situations of, say, trust negotiation where the interacting agents need to be cautious about the possible exposure of privacies during the communications. In below, we present an ontological model of interaction protocols based on the notion of finite state machine. We define an interaction protocol in terms of protocol name, the roles of participants, the underlying message systems, the rules that define the legal sequences of message for the interaction, and the goal of the protocol. We assume the existence of a class *IP-State-Slot* to express the variables of an interaction protocol and a class *Constraint* to express assertions about the variables.

$$\begin{aligned} \text{Interaction-protocol} \sqsubseteq & =1 \text{protocol-name} \text{String} \cap \forall \text{has-IP-role} \text{IP-Role} \cap \\ & =1 \text{has-message-system} \text{Message-System} \cap \forall \text{has-state-slot} \text{IP-State-Slot} \cap \\ & =1 \text{has-choreography} \text{FSM-Choreography} \cap =1 \text{has-goal} \text{Constraint} \end{aligned}$$

The choreography, rule of legal message sequences, is defined as a finite-state machine which consists of a set of state nodes and a set of transitions, with a distinguishing starting node.

$$\begin{aligned} \text{FSM-Choreography} \sqsubseteq & \forall \text{has-FSM-node} \text{FSM-Node} \cap \\ & =1 \text{has-FSM-starting-node} \text{FSM-Node} \cap \forall \text{has-FSM-Transition} \text{FSM-Transition} \\ \text{has-FSM-starting-node} \sqsubseteq & \text{has-FSM-node} \end{aligned}$$

A FSM-Transition is defined in terms of *from-node*, *to-node*, and optionally the *triggering-message* that triggers the transition, the *pre-condition* in which the transition is enabled, and *assertion* about the consequence of the transition.

$FSM\text{-}Transition \sqsubseteq =1from\text{-}node.FSM\text{-}Node \cap =1to\text{-}node.FSM\text{-}Node \cap \leq 1 triggering\text{-}message.Message \cap \leq 1pre\text{-}condition. Constraint \cap \leq 1 assertion. Constraint$

## 7 Conclusions

SW is a most promising technology to support VO applications. But there is still a gap between the state of art of SW and real world VO applications. Such gap is partly due to the conflicts between the insistency on some elegant formal properties of knowledge representation, such as decidability, and real-word requirements on adequate expressive power for systematic modeling. Without the introduction of multi-attribute constraints, the pure DL-representations of some concepts are not complete with respect to their consensus meaning. But the introduction of multi-attribute constraints generally implies the introduction of undecidability. We take an engineering viewpoint here and believe that a choice towards engineering solution would be a right way to push SW from academic enthusiasm towards widespread industrial acceptances. Aiming at VO applications, we propose a SW modeling language SWOCRL which allows the representation of object centered constraint rules. It is based on OWL and SWRL and specializes SWRL rules to the scope of single-class specification thus avoid combinatorial complexity of multi-class instances and gains reasoning scalability, while extends it with multi-attribute constraints for modeling expressibility. Some interesting VO concepts such as organizations, activities, contracts, interactions and their logistic specializations are described in OWL plus SWOCRL. Further work will investigate the issue of the modeling of complex interaction and interaction protocol reuse.

## References

1. Patel-Schneider, P. F., Hayes, P., and Horrocks, I. (2004). OWL web ontology language semantics and abstract syntax. Recommendation 10 February 2004, W3C.
2. RuleML. Rule markup language initiative, 2004. <http://www.ruleml.org>.
3. I. Horrocks, P.F. Patel-Schneider, H. Boley, S. Tabet, B. Groszof, and M. Dean. SWRL: A semantic Web Rule Language Combining OWL and RuleML. W3C Member Submission, May 2004. <http://www.w3.org/Submission/2004/SUBM-SWRL-20040521/>.
4. Jos de Bruijn, Rubén Lara, Axel Polleres, Dieter Fensel: OWL DL vs. OWL flight: conceptual modeling and reasoning for the semantic Web. In Proc. of the 14th international conference on World Wide Web, WWW 2005, Chiba, Japan, May 10-14, 2005.
5. Jeff Z. Pan and Ian Horrocks. Web Ontology Reasoning with Datatype Groups. In Proc. Of the 2003 International Semantic Web Conference (ISWC2003), pages 47–63, 2003.
6. P Gray, K Hui, & A Preece. Mobile Constraints for Semantic Web Applications. In M Musen, B Neumann, & R Studer (eds) Intelligent Information Processing, Kluwer, pages 117-128, 2002.

7. Chalmers, P Gray, & A Preece. Supporting Virtual Organisations using BDI Agents and Constraints, in M Klusch, S Ossowski, & O Shehory (eds) Cooperative Information Agents VI (LNAI 2446), Springer, pages 226-240, 2002.
8. Gruninger M (2003) PSL 2.0 Ontology – Current Theories and Extensions. <http://www.nist.gov/psl/psl-ontology/>
9. Paolucci, M. et al.: Semantic Matching of Web Services Capabilities. Proceedings of the 1st International Semantic Web Conference (ISWC 2002), Sardinia (Italy), Lecture Notes in Computer Science, Vol. 2342. Springer Verlag (2002)
10. Saraswat V A. Concurrent constraint programming[D]: Cambridge, MA,MIT Press,1993.

# Parameter Evolution for Quality of Service in Multimedia Networking\*

Ji Lu, Tao Li, and Xun Gong

Department of Computer Science, Sichuan University  
Sichuan Province, China  
dawangdaj@gmail.com

**Abstract.** Multi-path routing is an important mechanism for Quality of Service of multimedia communication in current networks. The key point of multi-path routing is to develop an efficient method to distribute the traffic over these multiple paths. This paper proposes a novel approach, QMP (QoS Multi-path Routing), to dynamically evolve the traffic distribution parameters using natural computation and make them adaptive to the changes of network flow and topology. First, a kind of function that reflects the changes of the network characteristics is coded using a novel natural computation technique. Then, regarding it as an antibody, QMP uses an artificial immune idea to evolve the function. Using the evolved function, some parameters in the networks can be predicted. After that, this paper proposes a traffic distribution approach that uses the parameters obtained to forward the real-time traffic, and in the meantime, adjusts the traffic parameters to adapt to the new network changes. The simulations show that this approach performs better than the classic optimal routing algorithms on feasibility and is also comparable to the optimal algorithms in terms of performance. Furthermore, QMP is better than the previous excellent multi-path and single-path routing approaches in both aspects of the delays and network resource usage.

## 1 Introduction

With the increase of multimedia communication in the current Internet, on the one hand, the requirements on the quality of the communication are becoming stricter; on the other hand, more applications running on the current networks lead to the more intensive competition on the limited network resources. So guaranteeing Quality of Service efficiently in multimedia networking becomes more and more important.

With the rapid increase of multimedia communication, people began to seek to use multiple paths and expected it could achieve good results. There has been a minimum-delay routing problem first described by Gallager [1]. And Gallager has also given an algorithm to this problem but it is unsuitable for practical networks.

---

\* This work was partially supported by the National Natural Science Foundation of China under the grant No. 60373110, 60573130 and 60502011, the New Century Excellent Expert Program of China Ministry of Education under the grant No. 04-0870, and the Innovation Foundation of Sichuan University under the grant No.2004CF10.

That is because the speed of convergence to the optimal routes in his algorithm depends on a global constant, and it requires that the input traffic and network topology be stationary or quasi-stationary, all of which can hardly be met in the present Internet. In this case, several algorithms have been proposed to date in order to improve Gallager's minimum-delay routing approach [2, 3, 4]. Segall and Sidi [3] use Merlin and Segall's method [5] to extend Gallager's minimum-delay routing algorithm and expect it to handle topological changes. Bertsekas and Gallager [6] use second derivatives to speed up convergence in Gallager's algorithm. However, all of these algorithms are also dependent on global constants and require that network traffic be static or quasi-static.

Cain et al. [4] have proposed a relatively good routing algorithm to reduce delays. But it requires that the update of routing-table at all routers should be synchronized, otherwise it would result in loops, which will increase the end-to-end delays. Because the synchronization intervals required by this algorithm must be known by all of the routers, this becomes another global reference that is also impractical in real-world networks. Furthermore, this approach is not scalable to large networks, because the time needed for synchronization of routing table update will become large, and this in return limits its responsiveness to short-term traffic fluctuations. Another drawback of this approach is lack of technique for asynchronous computation for multiple paths with instantaneous loop-freedom. In [7], an approximation to Gallager's method is proposed. But the computational complexity of traffic distribution in this approach is relatively high, and the detection of network topology and exchanges of update information among nodes must be very frequent, which consume too much network resource.

In this paper, a distributed approach, which is called QMP, is proposed to compute multiple paths for delivering large-volume real-time packets and to distribute traffic in networks using parameters obtained via natural computation. The rest of this paper is organized as follows: Section 2 first gives the formulation of the minimum-delay problem. Section 3 introduces some related work which has been done on multi-path routing. Section 4 presents the detail description of the proposed approach, QMP. Section 5 provides the simulations to compare the proposed approach with the previous methods. Finally, Section 6 concludes the whole paper.

## 2 Formulation of the Problem

In this section, the minimum-delay routing problem will be introduced. The minimum-delay routing problem was first formulated by Gallager [1]. A computer network  $G=(N, L)$  consists of  $N$  nodes and  $L$  edges among them. Each link is bi-directional with possibly different costs in each direction. Let  $L$  be the set of links. Let  $r_i(j)$  be the expected input traffic, which is measured in bits per second, entering the network at router  $i$ , and destined to router  $j$ . Let  $t_i(j)$  be the sum of  $r_i(j)$  and the traffic arriving from the neighbors of  $i$ , for destination  $j$ . And let traffic distribution parameter  $\phi_{ik}(j)$  denote the fraction of traffic  $t_i(j)$  that leaves router  $i$ , over the link  $(i, k)$ . Assuming that the network does not lose any packets, so it can be obtained that the following equation should hold for all  $i, j$ .

$$t_i(j) = r_i(j) + \sum_l t_l(j) \phi_{li}(j) \quad (1)$$

Let  $f_{ik}$  be the expected traffic on link  $(i, k)$ , measured in bits per second. The following equation can be obtained.

$$f_{ik} = \sum_j t_i(j) \phi_{li}(j) \quad (2)$$

The traffic distribution parameter set  $\phi$  should satisfy the following three conditions

$$\begin{cases} \phi_{ik}(j) = 0 & \text{if } (i, k) \notin L \vee i = j \\ \phi_{ik}(j) \geq 0 \\ \sum_k \phi_{ik}(j) = 1 \end{cases} \quad (3)$$

Let  $D_{ik}$  denote the expected number of messages per second transmitted on  $(i, k)$  times the expected delay per message. Assume the delay of messages depends only on the flow  $f_{ik}$  through  $(i, k)$ . So total expected delay per message times the total expected message number is given by the following expression.

$$D_T = \sum_{(i,k) \in L} D_{ik}(f_{ik}) \quad (4)$$

Therefore,  $D_T$  can be further expressed as a function of  $r$  and  $\phi$ .

So far, the minimum-delay routing problem can be expressed as follows.

For a given network topology, a set of input traffic flow  $r = \{r_i(j)\}$ , and delay  $D_{ik}(f_{ik})$  on each link  $(i, k)$ , the problem is to compute the traffic distribution parameter set  $\phi$ , such that total expected delay  $D_T$  can be minimized.

### 3 Related Work

Gallager derived the necessary and sufficient conditions, which should be satisfied to solve the problem above. This can be summarized as follows.

First, we can obtain the partial derivatives of the total delay  $D_T$  with respect to  $r$  and  $\phi$ .

$$\frac{\partial D_T}{\partial r_i(j)} = \sum_{(k,i) \in L} \phi_{ik}(j) \left[ D_{ik}'(f_{ik}) + \frac{\partial D_T}{\partial r_k(j)} \right] \quad (5)$$

$$\frac{\partial D_T}{\partial \phi_{ik}(j)} = t_i(j) \left[ D_{ik}'(f_{ik}) + \frac{\partial D_T}{\partial r_k(j)} \right] \tag{6}$$

where  $D_{ik}'(f_{ik}) = \partial D_{ik}(f_{ik}) / \partial f_{ik}$ . Let it be denoted by  $l_{ik}$ .

Then, the necessary condition and sufficient condition for a minimum of  $D_T$  with respect to  $\phi$  for all  $i \neq j$  and  $(i, k) \in L$  are respectively given in (7) (8).

$$\frac{\partial D_T}{\partial \phi_{ik}(j)} \begin{cases} = \lambda_{ij} & \phi_{ik}(j) > 0 \\ \geq \lambda_{ij} & \phi_{ik}(j) = 0 \end{cases} \tag{7}$$

$$D_{ik}'(f_{ik}) + \frac{\partial D_T}{\partial r_k(j)} \geq \frac{\partial D_T}{\partial r_i(j)} \tag{8}$$

Let  $S_i(j)$  be the set of neighbors of router  $i$  through which the traffic from  $i$  to  $j$  is forwarded.

And let  $D_i(j)$  denote  $\frac{\partial D_T}{\partial r_i(j)}$ .

For reviewing the problem in a distributed manner, we can reform the minimum-delay routing problem as follows: the solution of this problem is to discover the parameter set  $\phi$ ,  $D_i(j)$  and  $S_i(j)$  to satisfy the equation group (9).

$$\begin{cases} D_i(j) = \sum \phi_{ik}(j)(D_k(j) + l_{ik}) \\ S_i(j) = \{k \mid \phi_{ik}(j) > 0\} \\ D_i(j) \leq D_k(j) + l_{ik} \\ D_m(j) + l_{im} = D_n(j) + l_{in} & m, n \in S_i(j) \\ D_m(j) + l_{im} < D_n(j) + l_{in} & m \in S_i(j), n \in S_i(j) \end{cases} \tag{9}$$

In solving the problem, the determination of the set  $\phi$  is the key process. That is, how to distribute the traffic in these multiple paths directly influences the overall cost and performance. In particular, the traffic in networks changes dynamically. How to real-time distribute the traffic is very difficult to solve.

Some previous works have presented a few distributed solutions to this problem. Gallager [1] proposed a blocking technique to find the routes that are loop-free at first. However, this method have a significant drawback, that is, it depends on a



global step size  $\eta$ , which should be designated and used among all of the routers. But  $\eta$  is related to some specific input traffic pattern, which is not suitable for all routers to implement. Furthermore, to compute routing parameters using Gallager's method refers to a process of iteration to adjust parameters at every router, which requires that topology of the network should stay stationary long enough for completing the whole process of adjustment. So Gallager's method can be viewed as a theoretically good algorithm, rather than a feasible solution in practice.

In [7], a distributed algorithm is also proposed. It first uses link state routing algorithm to find multiple loop-free paths; then uses a heuristic to adjust traffic along the predefined multiple paths for each destination to approximate minimum delays. However, in this algorithm, queuing delays at every link must be measured and the update information of the whole network topology must be exchanged among the nodes periodically and very frequently. This consumes excessive bandwidth and would easily cause oscillations.

## 4 Description of QMP

### 4.1 Reformulation of the Problem

In the current Internet, the significant characteristic of the traffic is dynamic. To adapt to this feature, a good solution should be a distributed approach using local information to build the loop-free multiple paths, and able to dynamically adjust the distribution of network traffic on the routes to achieve high performance with low cost.

One major problem of implementing routing algorithms in the real network is the potential inconsistency of information distributedly stored in network nodes. To compute loop-free paths in such networks, the loop-free invariant [8] condition is first introduced as follows.

If the following two equations are always satisfied, any routing algorithm, regardless of its type, will lead to loop-free paths at every instant.

$$S_i(j) = \{k \mid D_{ki}(j) < F_i(j), (i, k) \in L\} \quad (10)$$

$$F_i(j) \leq D_k(j) \quad (k, i) \in L \quad (11)$$

where  $D_{ki}(j)$  is the value of  $D_k(j)$  reported to  $i$  by its neighbor  $k$ ;  $F_i(j)$  is the feasible distance from router  $i$  to  $j$  and is an estimate of  $D_i(j)$ . That is  $F_i(j)$  is equal to  $D_i(j)$  if the network topology information is convergent, but in other cases we could allow they are different temporarily.

It has been proved [8] that if the loop-free invariant conditions are met at any time  $t$ , the routing graph implied by  $S_i(j)$  is loop-free.

So far we can map Gallager's method, that is, the equation group (9) into (12)-(15).

$$\begin{cases} D_i(j) = \text{MIN}\{D_i(j) + l_{ik} \mid (i, k) \in L\} & (12) \\ S_i(j) = \{k \mid D_{ki}(j) < F_i(j), (i, k) \in L\} & (13) \\ F_i(j) \leq D_k(j) \quad (k, i) \in L & (14) \\ \phi_{ik}(j) = T((D_m(j) + l_{im}), \phi_{im}(j), k) & \\ \quad (m, i) \in L, (k, i) \in L & (15) \end{cases}$$

Equation (12) is employed for computing the shortest paths in a network. Equation (13) and (14), in fact implies a method to discover  $S_i(j)$  and could make sure the routing graph is loop-free. Equation (15) illustrates a function with the arguments  $(D_m(j) + l_{im}), \phi_{im}(j), k$ , responsible for distributing traffic in the network.

To compute  $D_i(j), S_i(j)$ , several exiting algorithms can be employed, such as ALP[10], OSPF[11]. However, efficient algorithms to distribute traffic at low cost are few.

In fact,  $T$  can be any function as long as it satisfies the conditions in (3). However, the mechanism of traffic distribution is directly related to the efficiency of routing and the overall delay. So the design of exact and efficient distributing approaches is needed, although challenging.

On the one hand, the parameters in function  $T$  are responsible for distributing traffic in any router in the network, which will change the flow on the routes. On the other hand, the continuously changing flow, in return, will lead to alternation or readjustment of the parameters. So the function  $T$  should be dynamically adaptive to the change of network flow.

In order to real-timely respond to the flow changes, it seems queuing delays at the links must be exactly measured periodically, and the update information must be exchanged among the routers very frequently. But this is hardly possible in the real-world networks if we really expect to make quick response.

#### 4.2 Using Computation Process Evolution (CPE) to Predict Network Parameters

Although traffic in the networks is changed dynamically, the distribution of flow in networks is predicable, as well as  $D_{ki}(j)$ , in a relatively short period of time  $r$  [13], especially in multimedia networking, in which the large volume of communication packets would generally last a long period of time  $R$  ( $R \gg r$ ) and the flows of data are almost continuous and of fixed direction.

In [9], Computation Process Evolution (CPE), an efficient natural computation approach is proposed. It aims at finding or predicting the computation process of any problem by only using a few input and output data, consisting of the cases needed to be satisfied and those needed to be avoided. It first encodes the antibody, which is an entity in artificial immunology, using a new natural computation technique. Through the gradual evolution, the affinity between antibody and the nonselves become more and more intense. At the same time, every time after the chromosomes are mutated,

the chromosomes should be checked to determine whether the antibody chromosome would match the selves, which are the conditions that should be satisfied.

Let  $D$  denote  $D_{ki}(j)$ .

Assume  $D = P(t)$ , that is,  $D$  is a function of time  $t$ .

Artificial immune method [14] is a kind of computation approaches, which solves a problem by distinguishing "self" and "nonself", and evolving the antibody progressively to eliminate the "nonself". CPE defines the antibody set as the computation process set  $P$ . Here, it is the set of functions of  $t$ .

Define the antigen set  $Ge$

$$Ge = \{(i, o) | i \in T, o \in S\}$$

where  $T$  is the set of time, and  $S$  is the set of  $D_{ki}(j)$ .

And  $Match$  is a relation on  $P \times Ge$

$$Match = \{ \langle p, g \rangle | p \in P, g \in Ge, p(g.i) = g.o \}$$

For some given samples  $(X_i, Y_i), (W_j, Z_j)$  the function  $P$  should be evolved out, such that

$$P(X_i) = Y_i \tag{16}$$

but

$$P(W_j) \neq Z_j \tag{17}$$

Non-self antigens are defined as  $(X_i, Y_i)$ . And self antigens are defined as  $(W_j, Z_j)$ .

The nonself  $(X_i, Y_i)$  and antibody  $P$  match if and only if  $P(X_i, Y_i)$  is equal to  $Y_i$  within a certain error. The self  $(W_j, Z_j)$  and antibody  $P$  match if and only if  $P(W_j, Z_j)$  is equal to  $Z_j$  within a certain error.

At first, QMP uses CPE to make exact prediction on the function  $P(t)$ , that is, the changes of  $D_{ki}(j)$ , according to the network changes based on only a few samples, and employs them to compute the traffic distribution parameters. Due to page limitation, the detail implementation of CPE [9] can not be described here.

### 4.3 Traffic Distribution

After  $P$  is discovered through the evolutionary process, at any moment of time  $t$  ( $0 \leq t \leq t_q$ ), where  $t_q$  is the maximal prediction lasting time,

$$D_{ki}(j) = P(t)$$

can be computed.

A novel traffic distribution algorithm TDA, using  $D_{ki}(j)$  obtained above is proposed in Fig. 1.

(1)  $\forall k \notin S_i(j), \phi_{ik}(j) \leftarrow 0;$   
 (2) **if**  $(|S_i(j)| = 1)$  **then**  $\forall k \in S_i(j), \phi_{ik}(j) \leftarrow 1;$   
 (3) **if**  $(|S_i(j)| > 1)$  **then**  $\phi_{ik}(j) \leftarrow \frac{1 - \frac{D_{ki}(j) + l_{ik}}{\sum_{g \in S_i(j)} (D_{gi}(j) + l_{ig})}}{S_i(j) - 1}$   
 $(k \in S_i(j));$   
 (4)  $D_{ki}(j) \leftarrow P(t);$   
 (5)  $\mathcal{E} \leftarrow \frac{\prod_{g \in S_i(j)} (D_{gi}(j) + l_{ig})}{\sum_{g, h \in S_i(j)} (D_{gi}(j) + l_{ig})(D_{hi}(j) + l_{ih})};$   
 (6)  $\forall k \in S_i(j), \phi_{ik}(j) \leftarrow \frac{\mathcal{E}}{D_{ki}(j) + l_{ik}}$

**Fig. 1.** Traffic Distribution Algorithm

**Theorem 2.** The Traffic Distribution Algorithm (TDA) satisfies the conditions in (3) at every instant.

*Proof.* Step 1 is able to guarantee  $\phi_{ik}(j) = 0$ , when  $(i, k) \notin L \vee i = j$

$$1 - \frac{D_{ki}(j) + l_{ik}}{\sum_{g \in S_i(j)} (D_{gi}(j) + l_{ig})}$$

According to  $\phi_{ik}(j) \leftarrow \frac{1 - \frac{D_{ki}(j) + l_{ik}}{\sum_{g \in S_i(j)} (D_{gi}(j) + l_{ig})}}{S_i(j) - 1}$  and  $\phi_{ik}(j) \leftarrow \frac{\mathcal{E}}{D_{ki}(j) + l_{ik}}$ ,

every numerator and denominator are nonnegative, so  $\phi_{ik}(j) \geq 0$ .

$$\sum_{k \in S_i(j)} \frac{\mathcal{E}}{D_{ki}(j) + l_{ik}} = \mathcal{E} \sum_{k \in S_i(j)} \frac{1}{D_{ki}(j) + l_{ik}} = 1, \text{ so } \sum_k \phi_{ik}(j) = 1.$$

The theorem holds.

This is very important for the distributing approach. Furthermore, according to the step 4-6, we can obtain the following theorem that is the guarantee of performance of this distribution policy.

**Theorem 3.** The volume of traffic designated to route  $(i, k)$  is in inverse proportion to  $D_{ki}(j) + l_{ik}$ . Furthermore,

$$\phi_{ik}(j) \cdot (D_{ki}(j) + l_{ik}) = \frac{\prod_{g \in S_i(j)} (D_{gi}(j) + l_{ig})}{\sum_{g, h \in S_i(j)} (D_{gi}(j) + l_{ig})(D_{hi}(j) + l_{ih})}$$

The approach can work efficiently because it adjusts the network traffic such that the routes with better conditions, fewer delays, deliver larger fraction of network traffic.

## 5 Simulation Results

### 5.1 Evaluation Methodology

In the simulations, random network topologies are generated, in which connectivity characteristics approximate real-world networks.

A random graph model is used which is similar to that of Doar [12] to resemble actual networks. In this experiment, a total of 600 nodes are randomly distributed in a Cartesian coordinate system. At first, the nodes are connected by a random spanning tree that is generated by iteratively adding a random edge between distinct fragments until all nodes in the network are combined into a single connected component. The existence of other edges are determined by the probability function in (18).

$$P_{(x,y)} = \beta \exp\left(\frac{-d_{x,y}}{2\alpha n}\right) \tag{18}$$

where  $d_{x,y}$  is the rectilinear distance between nodes  $x$  and  $y$ . The parameters  $\alpha$  and  $\beta$  govern the density of the graph. Increasing  $\alpha$  will result in the increase of the number of connections to nodes far away, and increasing  $\beta$  will result in the increase of the number of edges from each node. After some experiments,  $\alpha = 0.14$  and  $\beta = 0.24$  are chosen for generating the graphs which are considered proper to approximate the real-world networks.

### 5.2 Result of Comparisons

The optimal routing algorithm described by Gallager [1] is a classic algorithm, although not feasible in the real-world networks (referred as ‘‘OR’’). And the approach (‘‘DMP’’) proposed in [7] is also a good distributed one, the result of which is very close to the optimal algorithm and is already described above. The single-path routing

algorithm (“SPR”) is also taken into the comparison to measure the performance of multiple-path approaches. In order to ensure fairness and objectiveness, the proposed approach QMP, and compared methods run in the same environment. Every time, a random network is generated and the algorithms run on this network one by one.

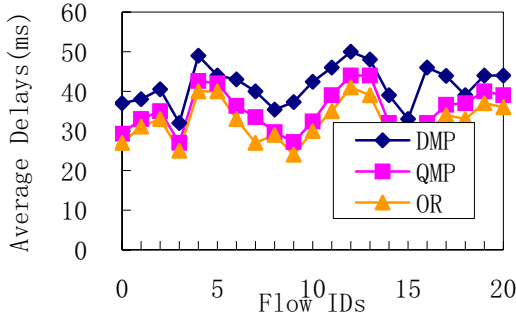


Fig. 2. Comparison on Average Delays among QMP, OR and DMP

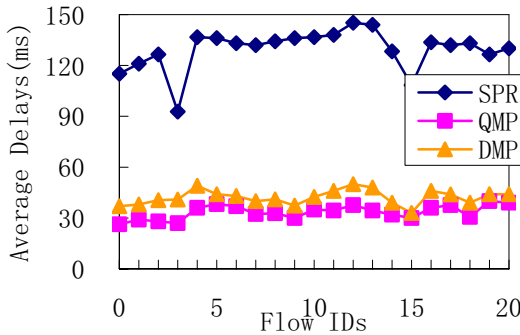


Fig. 3. Comparison on Average Delays among QMP, SPR and DMP

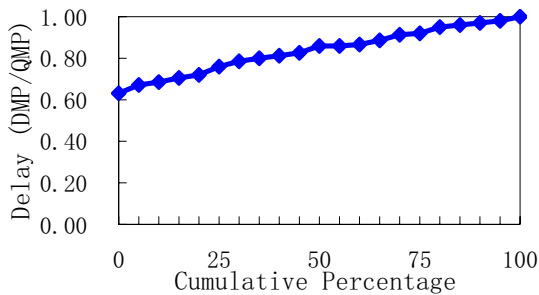
Fig. 2 gives the comparison on average delays among OR, DMP and QMP in the networks when the traffic is not too dynamic. There are 20 typical flows in each class chosen in this comparison. Assume OR can finally converge within an enough long period of time. The delays of QMP are smaller than those of DMP, and are more close to the result of the optimal routing algorithm. In this case, QMP and DMP are allowed to measure the network characteristics at the same frequency.

Although DMP ceaselessly measures the delays in the network and exchanges the updated topology information, these measurements and exchanges cannot be too frequent in the actual networks. Otherwise, only this kind of behavior will damage the normal communication and run out of the resource in the network. So in the real networks, DMP cannot do such network operations so frequently. However, not measuring the dynamic traffic real-timely leaves the DMP algorithm with worse results. And such differences between the QMP and DMP would be more obvious when operating at a lower frequency, which is already proved in the experiment.

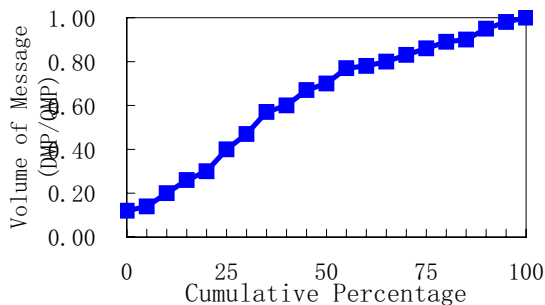
Fig. 3 shows the comparison on average delays among SPR, DMP and QMP. From the figure, it can be observed that the average delays of SPR are much larger than those of the multi-path routing approaches. In particular, the difference on the delays between QMP and SPR is more obvious if the networks are of high connectivity. Because in those situations, more chances of finding out fine paths are offered.

Because the performance of OR in highly dynamic traffic is very poor, the comparison in the condition of highly dynamic traffic is only made between DMP and QMP. In the same environment, both of DMP and QMP run 800 times. The average delays are computed and analyzed. Fig. 4 shows the ratio distribution of the average delays of QMP to that of DMP. It illustrates the cumulative percentage of cases in which the ratio is less than or equal to the value given on the abscissa. It can be observed that in the environment of highly dynamic traffic, QMP also performs better than DMP.

DMP and QMP need to measure the network traffic at some moments and exchange information among the neighbors. Fig. 5 shows the volume of message exchanged among the network nodes. Both of the two groups of observations are based on the condition that DMP and QMP result in the same delays. As the theoretical analysis mentioned above, the experiment also presents that DMP have to exchange much more update information of network topology than QMP. And this is one of the major disadvantages of this class of approaches that are similar to DMP, which prevents them from being widely used in the dynamic networks.



**Fig. 4.** Comparison on Average Delays between QMP and DMP



**Fig. 5.** Comparison on Volume of Messages between QMP and DMP

## 6 Conclusion

In this paper, an efficient mechanism for parameter determination and adjustment in multi-path routing with small delays is proposed. Unlike other approaches that have to ceaselessly measure the network changes and exchange much update information among all of the network nodes, the proposed approach in this paper uses only a few samples to predict the changes of some network characteristics in short periods of time. After evolution using natural computation, the parameters are employed in a novel traffic distribution approach to forward the dynamic traffic in the networks. The extensive simulations show that the performance of this approach is comparable to that of the Gallager's minimum-delay routing algorithm, and unlike the infeasibility of the latter, the proposed approach, QMP, can work in the real multimedia networks. And the comparisons also present that it renders smaller delays than previous excellent multi-path and single-path routing algorithms.

## References

1. R. G. Gallager: A Minimum Delay Routing Algorithm Using Distributed Computation. *IEEE Trans. Commun.* (1977)73–84
2. D. Bersekas and R. Gallager: Second Derivative Algorithm for Minimum Delay Distributed Routing in Networks. *IEEE Trans. Commun.* (1984)911–919
3. A. Segall: The Modeling of Adaptive Routing in Data Communication Networks. *IEEE Trans. Commun.* (1977)85–95
4. C.G. Cassandras, M.V. Abidi, and D. Towsley: Distributed Routing with Onn-Line Marginal Delay Estimation. *IEEE Trans. Commun.* (1990)348–359
5. P. M. Merlin and A. Segall: A Failsafe Distributed Routing Protocol. *IEEE Trans. Commun.* (1979)1280–1287
6. D. Bersekas and R. Gallager: Second Derivative Algorithm for Minimum Delay Distributed Routing in Networks. *IEEE Trans. Commun.* (1984)911–919
7. S. Vutukury and J.J. Garcia-Luna-Aceves: A Simple Approximation to Minimum-Delay Routing. *Proc. of ACM SIGCOMM*, Cambridge, MA ( 1999)
8. J.J. Garcia-Luna-Aceves: Loop-Free Routing Using Diffusing Computations. *IEEE/ACM Trans. Networking* (1993)130–141
9. Ji Lu: Computation Process Evolution. *Proc. of International Conference on Engineering of Intelligent Systems* (2006)
10. J.J. Garica-Luna-Aceves and M. Spohn: Scalable link-state internet routing. *Proc. International Conference on Network Protocols* (1998)
11. J. Moy: OSPF Version 2. RFC (1991)1247
12. M. Doar and I. Leslie: How bad is naive multicast routing? in *Proc. IEEE INFOCOM*, San Francisco, CA (1993)82-89
13. K. Loueset: Analysis of normal networking behavior in Internet. *Proc. International Conference on High Performance Networks* (2003)
14. Tao Li: *Compute immunology*. Publishing House of Electronics Industry, Beijing (2004)



# A DDL Based Formal Policy Representation

Maoguang Wang<sup>1,2,3</sup>, Li Zeng<sup>1,3</sup>, Jiewen Luo<sup>1,3</sup>, and Qing Yu<sup>1,3</sup>

<sup>1</sup> Institute of Computing Technology, Chinese Academy of Sciences, Beijing 100080, China

<sup>2</sup> School of Computer Science and Technology, China University of Mining and Technology, Xuzhou 221008, China

<sup>3</sup> Graduate School of the Chinese Academy of Sciences, Beijing 100039, China  
{wangmg, zengl, luojw, yuq}@ics.ict.ac.cn

**Abstract.** This paper introduces the policy uniform concept model and defines the general policy specification language-GPSL based on the formal foundation of DDL. Compared with other policy languages, GPSL describes the general concept model providing the support for the policies including action, goal, utility and so on. Moreover GPSL provides the support for complex action composition and description to define complex system behavior. To resolve the policy conflicts we use meta-policy to illustrate the policy relationships and owl to define policy ontology. Finally this paper analyzes the policy implementation in autonomic computing system.

**Keywords:** autonomic computing, DDL, GPSL(General policy specification language), policy ontology.

## 1 Introduction

The purpose of autonomic computing is to solve the software management crisis. Autonomic systems are expected to manage the increasing software complexity autonomously. Policy based management is the means by which humans will impose their will upon how autonomic systems govern themselves. Autonomic systems aim to provide a level of self-configuration, self-optimization, self-healing and self-protection that frees the system administrator from having to understand the changing complexities of the distributed IT systems and networks [1,2].

In autonomic computing the user can communicate with the autonomic computing system through the *commands* called *policy*. Policy is the important method for human-computer interaction and system management. That is, autonomic computing system can manage their behaviors based on the specified policy. Policy has many advantages:

- Policy separates the system control from the function: the separation of system behavior from action rules means that we change the system behaviors and need not reprogram or halt the system.
- Policy simplifies and automates the complex task management: the manager only specifies the needed policy and need not consider the concrete technique details so that it improves the management efficiency and scalability. Policy increases the

management level from the component to the system level, then to the business management level.

- Policy can reuse the expert knowledge efficiently. The management based on the policy embeds the intelligence to the operational system.

Therefore the policy research is very important and fundamental for the design and implementation of autonomic computing systems. Policy is applied in the network security before. Now people are becoming more and more concerned about the importance of policy application in the distributed, heterogeneous and complex system. However there are a lot of unsolved problems of policy theory and engineering. For instance it lacks the uniform definition of policy language. Also it lacks the efficient methods of policy application and policy implementation engineering etc.

The rest paper is organized as follows: Section 2 describes the related work of policy research in autonomic computing. Section 3 introduces policy formal foundation DDL action axioms and policy formal representation language GPSL. Section 4 presents the policy implementation in the autonomic computing system AGrIP. We finally present our conclusions in section 5.

## 2 Related Work

Now the research on policy focuses on the definition of policy language, the transformation and the engineering of policy.

Ponder[3] is a general policy specification language. It uses the subject to define the object of the domain, which also provides four control policies(Authorization, Information Filtering, Delegation and Refrain) and obligation policy. Ponder provides the mechanism of policy type definition for the reusability. Also it provides a series of policy aggregation (group, roles etc.) so as to define the complex policies. It also provides the meta-policy to resolve the policy conflict. Ponder language uses an object-oriented BNF language, which provides a toolkit to compile java or XML object for managing the policy life-cycle.

Rei[4] policy language defines the policy object including the action and condition, which connects the policy object to the subject. Rei provides four basic policies: Rights, Prohibitions, Obligations and Dispensations. It defines the semantics of Speech-Act(Delegation, Request, Revoke and Cancel). Rei provides a simple description of meta-policy, also defines the policy ontology using RDF/RDFS.

WSPL (Web Service Policy Language)[5] is the subset of OASIS eXtensible Access Control Markup Language(XACML). It defines the policies of authorization, QoS, reliable transfer and privacy, which also supports the combination of policies. WSPL defines the constraints of policies using standard data types.

Mandis et.al. [6] analyze the policy transformations(analysis model, online algorithm, simulation methods etc). Also it provides the methods based on machine learning.

IETF/DMTF[7] defines an object-oriented common information model, providing a QoS management framework based on policy, which is used to control the service quality. In this architecture, the main elements to control the policy is the Policy Enforcement Point(PEP) and Policy Decision Point(PDP), which manage the network using the policy console, policy library, PEP and PDP.

Role-based access control(RBAC) models[8] specify a policy interface for security administration, but do not provide guidelines for how large organizations manage their roles. Parameterized RBAC systems are even more expressive. However, this adds to the risk of dangerous mistakes during policy specification. Also it defines a formal model for hierarchical policy contexts: an RBAC meta-policy approach for subdividing the administration of large scale security environments and for enforcing information flow restrictions over policies.

The relevant works are very meaningful in autonomic computing, but lack a general efficient representation method for complex actions and policy combinations. We try to define a general specification language based on the formal logical foundation DDL and use owl to represent policy ontology in the real systems.

### 3 Policy Representation

There are many different policy definitions in different realms. For example, IETF/DMTF define policy as a series of management rules[7]. However policy is not equal to rules. Generally it is the method to guide system behaviors, which tells the autonomic element what to do(Goal), how to do(Action) and the ultimate effect(Utility). Policy language should have the uniform syntax, rich semantics and higher abstract levels. Also it needs flexible and extensible structure, rich representation ability and is easy to use.

#### 3.1 Formal Foundation DDL for Policy

Referring to the work[9,11], we use DDL as the formal logic foundation of policy. To represent knowledge and changes of agent in uniform way, an effective solution is to inherit the virtues of both description logics and action theories. Dynamic description logic (DDL for short) provides a uniform formalization framework for both static and dynamic representation and reasoning of knowledge. In DDL, domain axioms have three kinds of axioms, i.e., concept axiom, causality constraint axiom and action axiom. All of these axioms are closely related and useful for static reasoning and dynamic reasoning. Dynamic description logic is a suitable formalization modeling tools for dynamic domain. Considering the policy should be transferred to the executing action rules, we use the action axioms for action reasoning.

#### Action Axioms

Action axioms formalize how the world is changed. In dynamic domain changes are caused by actions. Two kinds of axioms are included, precondition axioms and succeeded state axioms, which can be easily obtained from corresponding action description. These two kinds of axioms can be expressed as the following, where  $A(x_1, \dots, x_n)$  is an action,  $u$  is the initial state,  $P_A$  is the preconditions of the action and  $E_A$  is the effects of the action:

- Precondition axiom:  $\text{Pre}(A(x_1, \dots, x_n), u) = P_A$
- Succeeded state axiom:  $\text{Suc}(A(x_1, \dots, x_n), u) = (u - P_A) \cup E_A$

For example, to move an object from one place to another place, this action can be described as follows:

$$(\text{move}(x, y, z) \equiv (\{\text{Block}(x), \text{Object}(y), \text{Object}(z), \text{On}(x, y), \text{On}(\text{null}, x), \text{On}(\text{null}, z)\}, \{\text{Block}(x), \text{Object}(y), \text{Object}(z), \text{On}(x, z), \text{On}(\text{null}, x), \text{On}(\text{null}, y)\}))$$

Thus the precondition axiom and succeeded state axiom can be shown as the following:

$$\begin{aligned} \text{Pre}(\text{move}(x, y, z), u) &= \{\text{Block}(x), \text{Object}(y), \text{Object}(z), \text{On}(x, y), \text{On}(\text{null}, x), \text{On}(\text{null}, z)\} \\ \text{Suc}(\text{move}(x, y, z), u) &= (u - \{\text{Block}(x), \text{Object}(y), \text{Object}(z), \text{On}(x, y), \text{On}(\text{null}, x), \\ &\text{On}(\text{null}, z)\}) \cup \{\text{Block}(x), \text{Object}(y), \text{Object}(z), \text{On}(x, z), \text{On}(\text{null}, x), \text{On}(\text{null}, y)\} \end{aligned}$$

Since the two kinds of action axioms formalize the character of dynamic changing of the world, the reasoning process is a kind of non-monotonic reasoning. These axioms show how to transform from one state to another state. But in most cases, action axioms are used together with other static axioms and inference rule can be shown as the following: The basic inference rule of DDL is MP rule (Modus Ponens), which can be denoted as:  $\varphi, \varphi \rightarrow \psi \Rightarrow \psi$

The MP rule can be used both on general formulas and assertion formulas.

To some extent, a action policy is considered triggerable when all its pre-conditions are satisfied, just as: IF {<condition\_1>...<condition\_n>} THEN {<action\_1>...<action\_k>}.

Considering the policy characteristic, it is very suitable to use DDL as the formal foundation of policy.

### 3.2 Policy Concept Model

We give a policy concept model referring to the work[10]. Supposing the autonomic system is in the state of  $s$  at time  $t$ ,  $s$  is usually described as a series of attributes and values. The policy guides the action  $a$  to execute directly or indirectly, thus leading the system to a new state  $s'$ . Therefore policy can be looked as a state-transition function. A policy  $P$  is four-tuple,  $P = \langle S_{\text{trigger}}, A, S_{\text{goal}}, U \rangle$ , where  $S_{\text{trigger}}$  is the set of states to trigger the policy execution;  $A$  is the set of actions to execute the policy;  $S_{\text{goal}}$  is the set of states after implementing the policy to achieve the goal;  $U$  is the function about  $S_{\text{goal}}$  called the utility function for evaluating the final result of the goals. Usually  $S_{\text{trigger}}$  is not empty. And we classify the policy into three basic kinds: action policy, goal policy, utility policy. Also we can mix the three policies to define more complex policies.

**Action Policy:** The policy  $P = \langle S_{\text{trigger}}, A, S_{\text{goal}}, U \rangle$  is an Action Policy if  $A$  is not empty while  $S_{\text{goal}}$  and  $U$  are empty. Action policy describes that the system should take actions in some certain states. The action policy does not specify the final state after taking actions, that is,  $S_{\text{goal}}$  is not specified.

**Goal Policy:** The policy  $P$  is a Goal Policy if  $S_{goal}$  is not empty while  $A$  and  $U$  are empty. It describes the states after the system achieves the goal. The goal policy describes the expected state and the system should select some action or a series of actions to take automatically. The manager only takes actions and need not know the underlying details. It is very flexible but needs more complex plan and model algorithms.

**Utility Policy:** The policy  $P$  is a Utility Policy if  $U$  is not empty while  $A$  and  $S_{goal}$  are empty. The Utility Policy evaluates the possible states so as to select the optimal state, which is the generalization of goal policy. Usually the Utility Policy is divided into some expectation value: Yes, No or some fuzzy description. To simplify it we specify it as 0, 1, or some certain value  $m \in (0, 1)$ . The utility policy needs more model, optimization and other relevant algorithms.

### 3.3 General Policy Specification Language(GPSL)

According to the policy concept model we define a general policy specification language(GPSL)[11] to represent the three policies.

#### Action Policy

$\langle \text{ActionPolicy} \rangle ::= \langle \text{Name} \rangle \langle \text{Performative} \rangle \langle \text{Type} \rangle \langle \text{Subject} \rangle \langle \text{Object} \rangle \langle \text{Action} \rangle \langle \text{Precondition} \rangle \langle \text{ConstraintLanguage} \rangle \langle \text{Duration} \rangle \langle \text{Priority} \rangle$ , where  $\langle \text{Name} \rangle$  is the policy identifier,  $\langle \text{Performative} \rangle$  represents the policy operation performatives such as Authorization, Prohibition, Delegation and so on.  $\langle \text{Type} \rangle$  represents two values: true or false which illustrate the positive and negative policy respectfully.  $\langle \text{Subject} \rangle$  is the policy executor including agent, group or role etc.  $\langle \text{Object} \rangle$  is the action operator.  $\langle \text{Precondition} \rangle$  specifies the policy trigger state which is usually described by first order predicate logic.  $\langle \text{Action} \rangle$  is the action or the set of policy actions.  $\langle \text{ConstraintLanguage} \rangle$  specifies the state constraint language.  $\langle \text{Duration} \rangle$  illustrates the period of validity. Its default value is 0 which means it has no limit in the whole lifecycle.  $\langle \text{Priority} \rangle$  refers to the priority of policy which is usually used to resolve the policy conflict. We can specify the priority discrete value  $p \in [-n, +n], n > 0$ . Further we can divide the action policy into three policies: Authorization Policy, Delegation Policy and Prohibition Policy.

As to the Action policy, the policy is an authorization policy if the performative is authorize. It states whether the subject can take the action to the object. If  $\langle \text{type} \rangle$  is true it means that the subject can take actions otherwise the subject can not. The negative policy provides more flexibility. We can use negative authorization policy to cancel the actions taken by the subject. Given the policy (**:name** AuthorizePolicy1 **:performative** "Authorize" **:subject** spideragent1 **:object** dbagent1 **:action** update **:precondition** withState(agent1, ready)), it shows that when spideragent1 works spideragent1 can update dbagent1.

The action policy is a delegation policy if the performative is delegate which specifies the rights owned by the subject. The delegation policy permits the subject grant its rights to the target object so that the target object can take some operations.

The delegation policy is usually used to transfer the access rights. Given the policy (**:name** DelegationPolicy1 **:performative** “Delegation” **:subject** spideragent1 **:object** spideragent2 **:action** delete **:precondition** hasRight(spideragent1, dbagent1, delete)), it shows that when spideragent1 takes action1 it can delegate the right to spideragent2.

The action policy is a prohibition policy if the performative is prohibit. It illustrates that the subject is prohibited to take actions on the target object under some conditions. The difference between the prohibition policy and the negative authorization policy is that the former is executed by the subject policy controller while the latter is executed by the object policy controller. For instance, one policy (**:name** ProhibitPolicy1 **:performative** “Prohibit” **:subject** spideragent1 **:object** dbagent1 **:action** action2 **:precondition** withState(spideragent1, busy)) , shows that the spideragent1 is prohibited to take action2 to dbagent1. The policy controller of spideragent1 will monitor the execution.

The action policy is an obligation policy if the performative is obligate where the subject and the object are usually same. It controls the system function behaviors and provides the capability to response to the dynamic environment while access control policy controls the right of the system components. Given the policy (**:name** RestartPolicy **:performative** “Obligate” **:subject** spideragent1 **:action** restart **:precondition** lowerthan(availablememory, 1M)), it illustrates that when the memory is lower than 1M spideragent1 can take action restart.

### Goal Policy

$\langle \text{GoalPolicy} \rangle ::= \langle \text{Name} \rangle \langle \text{Performative} \rangle \langle \text{Subject} \rangle \langle \text{Object} \rangle \langle \text{Precondition} \rangle \langle \text{Postcondition} \rangle \langle \text{ConstraintLanauge} \rangle \langle \text{Duration} \rangle \langle \text{Priority} \rangle$ , where  $\langle \text{Postcondition} \rangle$  illustrates the final state,  $\langle \text{Performative} \rangle$ 's value is “achieve”. Compared with action policy their difference is that  $\langle \text{Action} \rangle$  describes how to do while  $\langle \text{Postcondition} \rangle$  shows that what to do. Given the policy (**:name** goalpolicy1 **:performative** “Achieve” **:subject** ftpagent **:precondition** between(clienthost, 192.168.0.1, 192.168.0.100) **:postcondition** greater(bandwidth(client), 1M)), it shows that if the client IP address is between 192.168.0.0 and 192.168.0.0.100 ftpagent should allocate the client the bandwidth more than 1M. Of course ftpagent should adopt the planning method to choose suitable action sequences to achieve the goal.

### Utility Policy

$\langle \text{UtilityPolicy} \rangle ::= \langle \text{Name} \rangle \langle \text{Performative} \rangle \langle \text{Subject} \rangle \langle \text{Object} \rangle \langle \text{Precondition} \rangle \langle \text{UtilityFunction} \rangle \langle \text{ConstraintLanauge} \rangle \langle \text{Duration} \rangle \langle \text{Priority} \rangle$ , where  $\langle \text{UtilityFunction} \rangle$  is a function relevant to the target state.  $\langle \text{Performative} \rangle$ 's value is “Optimize”.

From the syntax the difference among the three policies is that  $\langle \text{Action} \rangle$  in action policy illustrates how to do,  $\langle \text{Postcondition} \rangle$  in goal policy shows what to do while  $\langle \text{UtilityFunction} \rangle$  describes the final effect. Given the policy (**:name** utilitypolicy1 **:performative** “Optimize” **:subject** httpagent **:utilityfunction**

$f = w_1 \sum_{i=1}^n \text{band\_width}(client_i) + w_2 \sum_{i=1}^n \text{response\_time}(client_i)$  ), it shows that

httpagent should ensure the optimal balance between the bandwidth utility and the client response time according to the utility function and system model. After that it will plan the action sequences to achieve the goal.

## Actions

In the policies the action is very important because finally the goal and the utility function should be transferred to the executing actions. And we divide the actions into two kinds: Atomic Action and Composite Action. Atomic action refers to the concrete operation:  $\langle \text{AtomicAction} \rangle ::= \langle \text{Name} \rangle \langle \text{Target} \rangle \langle \text{Operation} \rangle$ , where  $\langle \text{Name} \rangle$  is the identifier,  $\langle \text{Target} \rangle$  is the object to take actions.  $\langle \text{Operation} \rangle$  refers to the target methods.

Complex actions can be divided into cycle action, sequential action, parallel action, optional action and finite state machine action etc. Cycle action shows a sub-action's recurrence. Sequential actions consist of sub-actions and execute in the sequential orders. Optional actions illustrates that it will choose a sub-action under a specified condition. FSM action is the generalization of other actions, whose sub-action will execute according to the defined FSM. We can see that the composite action just schedule its sub-actions according to the policy. Given the policy `action1 = (:name action1 :operation disconnect()); action2 = (:name action2 :operation start("Rising")); action3 = (:name action3 :operation start("Ghost")); action4 = (:name action4 :operation connect()); action5 = ParallelAction(action2, action3); killLovegateAction = SequentialAction(action1, action5, action4)`, it shows that when the system finds the virus(Lovegate etc.) it should disconnect the network and start the software to kill the virus(Rising), the backup software(Ghost). After they complete the work they will restore the network connect. Apparently this action is a complex action. Thus we can make the user to specify the policy conveniently illustrating the complex actions to define the system behaviors.

## 4 Policy Implementation

### 4.1 Policy Conflict Resolution

When there are many policies it maybe cause the policy conflict. For example  $p1$  authorize agent1 the right to access the database while  $p2$  prohibit agent1 to do that. We classify the policy conflicts into two kinds: Modality Conflicts represent that the subject take the same actions to the same object while performatives are conflict(Positive Obligation vs. Negative Obligation). Another conflict is the application conflict. For example the policy  $p3$  and  $p4$  simultaneous specify agent2 and agent3 use the same resource  $r$ . However  $r$  only has one user exclusively.

Meta-policy is the policy about policy which stipulate the policy relationships. Meta-policy restrains the policy and can resolve the policy conflict. To the modality conflicts meta-policy prescribes that one performative has more priority than the other performative. For instance meta-policy can specify Prohibition has higher priority than Authorization. Therefore when the two policy conflict occur the Prohibition policy will execute.

Using meta-policy to define the policy relationship and policy ontology to resolve the policy transformation, figure 1 depicts the policy ontology.

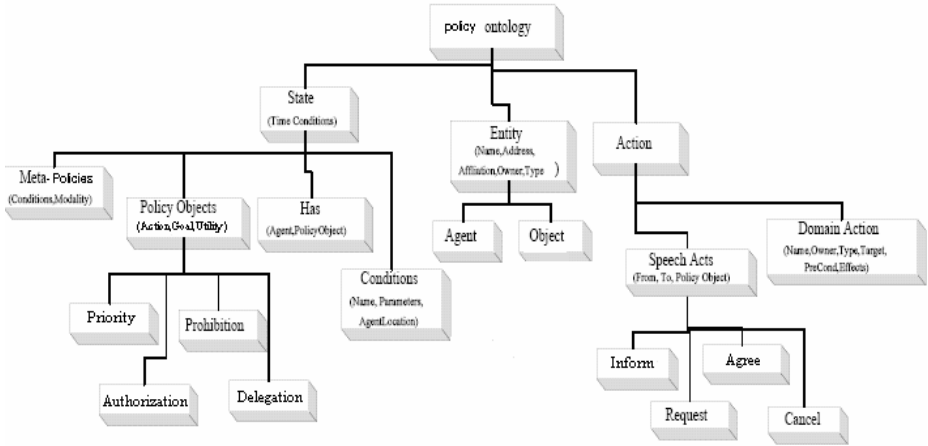


Fig. 1. Policy ontology

```

<owl:Class rdf:ID="Policy">
  <rdfs:comment>
    A policy is a constraint on the behavior of agents.
  </rdfs:comment>
  <rdfs:label>Policy</rdfs:label>
</owl:Class>
<owl:ObjectProperty rdf:ID="constraints_action">
  <rdfs:comment>
    a policy constraints the actions performed by an agent.
  </rdfs:comment>
  <rdfs:label>constraints</rdfs:label>
  <rdfs:domain rdf:resource="#Policy"/>
  <rdfs:range rdf:resource="#Action"/>
</owl:ObjectProperty>
<owl:Class rdf:ID="Obligation">
  <rdfs:comment>An obligation is a kind of policy that
prescribes actions and/or states of an agent and/or
resource.
  </rdfs:comment>
  <rdfs:label>Obligation</rdfs:label>
  <rdfs:comment>

```



```

    An obligation of a type of action policy
  </rdfs:comment>
  <rdfs:subClassOf rdf:resource="#Policy"/>
</owl:Class>

```

In the autonomic computing the specified policy can be ultimately transferred to XML:

```

<policy_set name="agentID_policy_library"
id="policy_lib1" ...>
  <Policy name="RestartPolicy" id="Restart1"
subject="spideragent1" ...>
    <policy_rule>
      <trigger>computer_busy </trigger>
      <condition>
        <parameter>memory </parameter>
        <operator>lower </operator>
        <value>1M</value>
      </condition>
      <action name="restart"><operation>
restart</operation></action>
    </Policy rule>
  </policy>
</policy_set>

```

## 4.2 Policy Application

AGrIP(agent grid intelligence platform) using policy is developed on the basis of MAGE[14]. Our final goal is to make full use of agent and autonomic computing techniques to build an autonomic grid environment for the industry applications. In the past decade attempts to apply intelligent agents to realize the grid vision have been made by academic researchers. The most interesting work in the literature might be the Agent Grid concept proposed under the DARPA ISO's Control of Agent-Based Systems (CoABS) program. And we can view the grid as a number of interacting agents. Base on our previous work[11,12,13] we are trying to devise an autonomic grid intelligence platform. Because autonomic computing is an evolutionary process AGrIP also evolves from the agent grid intelligent platform to autonomic grid platform. Further work will be introduced detailedly and please refer to the website <http://www.intsci.ac.cn>[14]. Figure 1 shows the AGrIP fundamental layered architecture.

AGrIP makes it possible to configure autonomic element based on agent dynamically through deployment solution center(DSC). Using address mapping

facilitator(AMF) autonomic element can communicate with name directly so as to eliminate the address complexity. Using directory facilitator(DF) we can provide self-configuring and self-optimizing based on service. Broker has the abilities of service composition and coordination. While policy management center(PMC) is responsible for the policy management. PMC usually includes the functions: policy editor, policy generation and definition based on ontologies, policy analysis and resolution. MTS provides the transparent communication. Security facilitator(SF) answers for security detection and recovery.

AGrIP can analyze hardware utilization to decompose the specified task to the relative idle nodes dynamically according to the policy rules specified by PMC. For instance, if the running node is busy it will select the relative idle node according to the specified policy. Then the task is transferred to the idle node. It has been applied successfully in a complex, dynamic, multi-agent domain, which is a promising approach to some of the challenges in autonomic computing initiative for they need to be able to adjust their behaviors to changing operating conditions and malfunctioning components based on policies.

## 5 Conclusions

Our research aims at the policy representation based on DDL. Compared with other policy languages, GPSL has the general concept models providing the support for the complex policy definitions and compositions. To resolve the policy conflict and policy transformation we define the policy ontology using OWL. The policy will finally be translated into the XML policy rule illustration. Further we are doing research on policy consistency checking and apply it to the AGrIP.

## Acknowledgment

This work is supported by the National High-Tech Programme of China (Grant No. 2003AA115220), the National Basic Research and Development Plan of China (Grant No. 2003CB317000) and the Youth Research Programme of China University of Mining and Technology (Grant No. 0D4489).

## References

1. Jeffery O. Kephart, David M. Chess. The Vision of Autonomic Computing Outlook. IEEE Computer Society, January 2003, pp.41-47.
2. A.G.Ganek, T.A.Corbi. The dawning of the autonomic computing era. IBM SYTEMS JOURNAL, VOL42, NO 1,2003, pp.5-18.
3. Nicodemos Damianou, Naranker Dulay, Emil Lupu, and Morris Sloman. The Ponder policy specification language. in Policy 2001: Workshop on Policies for Distributed Systems and Networks, Bristol, UK, Jan. 2001, pp. 18--39.
4. Lalana Kagal. Rei : A Policy Language for the Me-Centric Project. TechReport, HP Labs, September 2002.
5. Anne Anderson. An Introduction to the Web Services Policy Language. 5th IEEE International Workshop on Policies for Distributed Systems and Networks (POLICY 2004).

6. Mandis Beigi, Seraphin Calo, Dinesh Verma. Policy Transformation Techniques in Policy-based Systems Management. 5th IEEE International Workshop on Policies for Distributed Systems and Networks (POLICY 2004).
7. Distributed Management Task Force, Inc. (DMTF). Common Information Model (CIM) Specification, version 2.2, available from <http://www.dmtf.org/spec/cims.html>, June 14, 1999.
8. Sandhu, R.S., E.J. Coyne, H.L. Feinstein, and C.E. Youman. Role-Based Access Control Models. *IEEE Computer*, 1996. 29(2): p. 38-47.
9. Zhongzhi Shi, Mingkai Dong, Yuncheng Jiang, Haijun Zhang. A Logical Foundation for the Semantic Web. *Science in China Ser. F Information Sciences* 2005 Vol.48 No.2 161-178.
10. Jeffrey O. Kephart, William E. Walsh. An Artificial Intelligence Perspective on Autonomic Computing Policies. 5th IEEE International Workshop on Policies for Distributed Systems and Networks (POLICY 2004): 3-12
11. Haijun Zhang. Research On Agent-Based Autonomic Computing. PhD Dissertation. Chinese Academy of Sciences, Beijing, 2005.
12. Maoguang Wang, Zhongzhi Shi, Jiewen Luo, et al. Dynamic Interaction Protocol Load in Multi-agent System Collaboration. *Proceedings of the 8th Pacific-Rim International Workshop on Multi-Agents.2005*,pp.101-115.
13. Maoguang Wang, Jiewen Luo, Fen Lin, Zhongzhi Shi. Internet intelligence Platform-AGrIP. *Artificial Intelligence Applications and Innovations II(AIAI2005)*, 2005,pp.363-370.
14. Intelligent Science Web site: [http://www.intsci.ac.cn\[OL\]](http://www.intsci.ac.cn[OL]), 2006.

# Concurrent Agent Social Strategy Diffusion with the Unification Trend

Yichuan Jiang and Toru Ishida

Department of Social Informatics, Kyoto University, Japan  
jiangyichuan@yahoo.com.cn, ishida@i.kyoto-u.ac.jp

**Abstract.** In massive agent system, there are many diffusion processes among agent social strategies which take place concurrently. A social law is a restriction on the set of strategies available to agents [1]. All agents will trend to select an identical social strategy in the agent social law evolution, which can be called as the phenomenon of *unification trend*. This paper presents a model for the concurrent social strategy diffusion with unification trend. With the model, an agent's social strategy is influenced not only by the diffusion that bear on itself, but also by concurrent diffusion processes that bear on other agents; and, an agent will incline to the average social strategy of the whole system which can make the system be more unified.

**Keywords:** agent, social law, social strategy, diffusion, concurrent mechanism.

## 1 Introduction

The concept of artificial social system provides a new way to make coordination among agents, whose basic idea is to add a mechanism, called a *social law*, into the system [2][3][4]. According to [1], social law is the set of restrictions for agents' activities which allow them enough freedom on the one hand, but at the same time constrain them so that they will not interfere with each other [5].

Each agent has its own social strategy, and those social strategies among different agents may bring out collisions. To receive the harmony of the system, we need to endow a social law to the system, and the social law is the set of the social strategies that can be accepted by all agents. If each agent obeys the social law, then the system can avoid collisions without any need for a central arbiter [1]. Moshe Tennenholtz etc. have made some benchmark researches for the construction of social law [1][2][3]. However, they have not yet made systematic research about the evolution from individual social strategies to the global social law by the agents' interactions, and they often think the design of social law is off-line [2].

Then, how can we get the global social law? One answer is to admit that agents will have individual social strategies, but the social law can be evolved from the diffusions of social strategies [6][7].

In [6], we initiated a study on how to evolve individual social strategies to the social law by the hierarchical immediate agent diffusion interaction. Our initial model

in [6] mainly considers how the superior agents diffuse their social strategies to the junior agents, and the diffusion only take places with the hierarchical immediate form. However, in real society, there are so many collective intentions and practices [8], and the social strategies shared by many junior agents can also influence the superior agents. Therefore, we also presented a new collective diffusion model in [7]. By the model, we can see that the social strategy of a superior agent may be changed by other social strategy owned by collective agents.

Our agent social law evolution model in [6] is the diffusion form of one by one, and the model in [7] is based on the basic diffuse form from collective agents to one agent. The two models are both *sequential*, and don't consider the *concurrent* diffusion processes among many agents. In the real society, there are many diffusion processes from collective agents to collective agents which take place concurrently. Therefore, we need to explore the concurrent mechanism in the agent social law evolution.

Otherwise, in the concurrent agent social law evolution, there is an interesting phenomenon which can be called *unification trend* in this paper. When many diffusion processes take place concurrently in the agent system, the agents will try to select an identical average social strategy which can make the system more unified. Such phenomenon is also familiar in real society where people always tend to select a general social strategy which will make the community be more harmonious.

In this paper, we will mainly address the concurrent social strategy diffusion, and explore the unification trend in the diffusion. Our model is explained by the instance of people queue exercitation.

## 2 Illustration for Our Instance

### 2.1 Our Instance

As like [7], in this paper, we also take the people standing orientations in the queue exercitation as an example to explain our model. In the system, each agent can stand with its orientation, and our goal is to minimize the total number of agent orientations. In the soldiers queue exercitation, each soldier can stand with his orientation, but they will keep their unification of standing orientation while they make queue exercitation. Therefore, at the initial stage of agent system, each agent stands with its orientation, and after continuous diffusion, we will try to make the agents stand with a unique orientation. If many agents or some superior agents stand for an orientation, then the other many individuals had no choice but to fall back on the strong orientation.

In our agent system, each agent can stand with one of the 8 orientations, seen as Fig 1. Therefore, the social strategy of an agent is its orientation. Let  $n$  be the number of agents, we can use an array to denote the social strategies of agents.  $L[i] \rightarrow \{1, \dots, 8\}, 1 \leq i \leq n$ , represents social strategy of agent  $i$ , *i.e.* the standing orientation of agent  $i$ . For example, in Fig.2, the social strategies of the agent system in (a) is shown as (b).

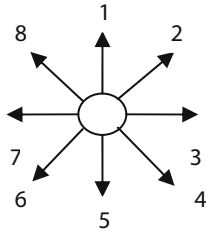


Fig. 1. 8 social strategies of agent

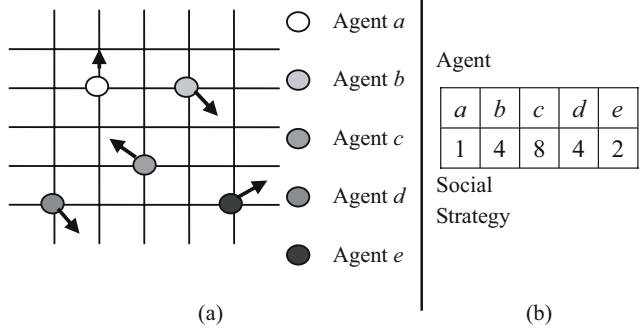


Fig. 2. An agent system and its social strategies

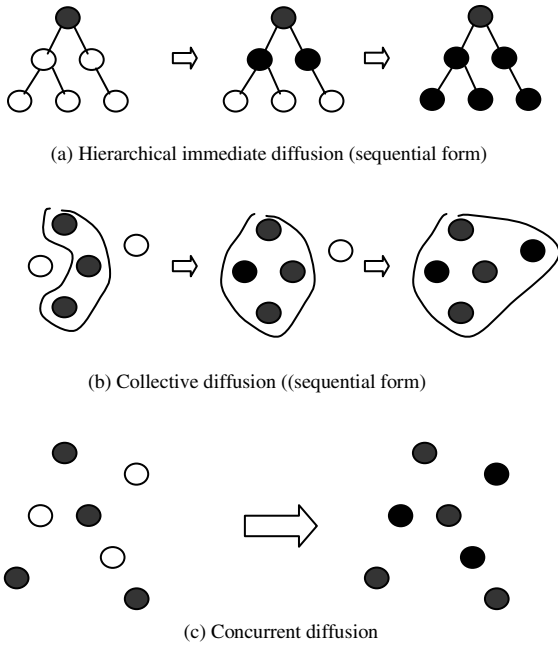
### 2.2 The Concurrent Mechanism and Unification Trend

In our previous works about agent social law evolution, the social strategy diffusion takes place as the sequential form. However, the diffusion is always concurrent in real society which means that many diffusion processes among agents may take place simultaneously. The concurrent mechanism of social law evolution includes the simultaneous diffusion with the form of one to one, collective to one, one to collective, and collective to collective. Fig 3 is the three forms of agent diffusion models. We mainly explored the hierarchical immediate diffusion in [6], and the collective diffusion in [7]. Now we will explore the concurrent diffusion in this paper.

From Fig.3 (c), we can see that the many diffusion processes in the system may take place concurrently. When we consider a diffusion process, we need to consider the impact of other diffusion processes that take place concurrently. Let  $diffusion(i)$  be the diffusion that bear on agent  $i$  and  $A$  be the agent set, the social strategy of agent  $a$  is not only determined by  $diffusion(a)$ , but also determined by  $\bigcup_{b \in A-a} diffusion(b)$ .

When many diffusion processes take place concurrently in the agent system, the agents will try to select an identical average social strategy, which can be called as the *unification trend*. In the society, the trend toward unification of the social strategies is obviously important in determining the evolution of the social law. With the trend to unification, the agents will incline to select the average social strategy, which can make the conflicts of the system be minimized. Finally, the social strategies within the system will go toward to unification. The evolution of social law is a collective and concurrent phenomenon, affecting by its collective and concurrent nature the way that individual agents and groups within a system think and act. In the social law evolution, the individual agents will incline to select an identical strategy which can make the system be more harmonious.

In the concurrent agent social law evolution, the diffusion of social strategies takes place under the concurrent form with the trends to unification of social strategies. Therefore, the social strategy of an agent is determined not only by the diffusion



**Fig. 3.** Three forms of agent social strategy diffusion

bearing on itself, but also by other diffusions that bearing on other agents; the trend to unification of an agent is also determined not only by itself, but also the unification trend progress of other agents.

### 2.3 Related Concepts in Our Instance

In the agent system, each strategy has its authority. If a social strategy is accepted by many agents, its authority will be strong. However, different agents may contribute differently to the authority of a social strategy. Obviously, the superior agents will contribute to the social strategy’s authority more than the junior agents. The authority of a social strategy in the system can be called its *rank*. The concept of *rank* is always used in the web search [9], which denotes the importance of web pages. Here, we use this concept to represent the authority of a social strategy.

**Definition 1.** Social position of agent  $i$  can be a function:  $p_i \rightarrow [0, \vartheta]$ , where  $\vartheta$  is a natural number.  $p_i > p_j \Rightarrow$  the position of  $i$  is superior to  $j$ . The set of the positions of all agents in the system can be denoted as:  $P = [p_i]$ , where  $1 \leq i \leq n$ , and  $n$  denotes the number of agents in the system.

In this paper, we think that: if agent  $a$  has a social strategy  $l$ ,  $a$  is implicitly conferring some importance to  $l$ . Then, how much importance does an agent  $a$  confer to its social strategy  $l$ ? Let  $N_a$  be the number of social strategies of  $a$  and let  $p_a$  represent the social position of agent  $a$ , the agent  $a$  confers  $p_a / N_a$  units of rank to  $l$ . In our instance, the number of social strategies of an agent is only  $1$ , therefore, the agent  $a$  confers  $p_a$  units of rank to  $l$ .

**Definition 2.** We may adopt the matrix to denote the authority of each social strategy. If agent  $i$  has the social strategy  $j$ , then let the matrix entry  $m_{ij}$  have the value  $p_i / N_i$ , or else the matrix entry  $m_{ij}$  have the value  $0$ . Therefore, let  $A$  be the set of agents in the system, the authority of social strategy  $j$  can be defined as:

$$\forall_j Rank(j) = \sum_{i \in A} m_{ij} \tag{1}$$

A social strategy may be accepted by many agents which can be classified as a group. For example, agent  $b$  and  $d$  both have the same social strategy  $4$ .

**Definition 3.** Now, we will define the *overlay group of a social strategy*. Let  $G(l)$  be represented as the overlay group of social strategy  $l$ , we have:

$$\forall_l G(l) = \{u \mid \text{agent } u \text{ adopts the social strategy } l\} \tag{2}$$

Obviously, the authority of a social strategy is determined by the members of its overlay group. Therefore, Let  $G(l)$  be the overlay group of social strategy  $l$ ,  $N_u$  be the number of social strategies of agent  $u$ , then the authority of social strategy  $l$  in Equation (1) can also be defined as:

$$\forall_l Rank(l) = \sum_{u \in G_l} p_u / N_u \tag{3}$$

Agent can only have one social strategy in our instance, so the authority of social strategy  $l$  can be simplified as:

$$\forall_l Rank(l) = \sum_{u \in G_l} p_u \tag{4}$$

We will use the concept of group authority to represent the social strategy authority, *i.e.*  $\forall_l Rank(G(l)) = Rank(l)$ .

**Definition 4.** The difference of two social strategies  $i$  and  $j$  can be defined as the distance of their represented standing orientation:

$$DL_{ij} = \begin{cases} |j - i|, & \text{if } |j - i| \leq 4 \\ 8 - |j - i|, & \text{if } |j - i| > 4 \end{cases} \tag{5}$$

### 3 Diffusion Coordination Among Inclined Social Strategies

When diffusion processes take place concurrently with the unification trend, each agent will incline to select the social strategy with strong impact strength as well as



the social strategy that that will accepted by other agents. Therefore, when we decide the change probability of an agent’s social strategy, we will consider the *social strategy impact strength* and the *average social strategy* of all agents together.

In the diffusion, agent will go toward to a new social strategy which can be called as *inclined social strategy*. In the following part, we will consider the impact force of different inclined social strategies to an agent by considering the concurrent mechanism and unification trend in the diffusion.

### 3.1 Trend to Strong Social Strategy

The impact strength of a social strategy at a place will be decided by the collective positions of the agents within its overlay group, and the geographical distribution. Let  $s_l(x, y)$  denotes the impact strength of social strategy  $l$  at the place of  $(x, y)$ ,  $i \in \Theta(x, y)$  denotes that agent  $i$  doesn’t locate at the place of  $(x, y)$ , then  $s_l(x, y)$  is defined as:

$$s_l(x, y) = \sum_{i \in G(l), i \notin \Theta(x, y)} \frac{\alpha_1 p_i}{\alpha_2 \cdot \sqrt{(x_i - x)^2 + (y_i - y)^2}} \tag{6}$$

Where  $\alpha_1$  and  $\alpha_2$  are parameters to determine the relative importance of the two factors. If agent  $a$  is located at the place of  $(x, y)$ , the set of social strategies in the system is  $L$ , then the probability that  $a$  select social strategy  $l$  in the diffusion can be initially defined as:

$$prob_i^l = s_l(x, y) / \sum_{m \in L} s_m(x, y) \tag{7}$$

Obviously, the above definition for change probability in diffusion is simple, and which doesn’t consider the concurrent mechanism and unification trend in the diffusion. When an agent selects a social strategy according to the above definition, it will only consider the impact strength of the social strategy, but not consider the concurrent diffusion processes of other agents, and not trend to go toward to a unified social strategy of the system.

### 3.2 Concurrent Trend to Current Average Social Strategy

The diffusions of all social strategies take place concurrently, and the agents tend to select an identical social strategy that will be accepted by all agents. Therefore, the agent will incline to select not only the social strategy with strong impact strength, but also the average social strategy which can satisfy the trend to unification.

Then, what is the average social strategy? Obviously, it is not the simple average value of all social strategies. Therefore, let  $L$  be the set of the social strategies in the system, the *current average social strategy of the system (CASS)* can be defined as:

$$CASS \approx \sum_{l \in L} \frac{rank(l)}{\sum_{m \in L} rank(m)} \cdot l \tag{8}$$

The inclination of an agent will compromise the two factors of *social strategy impact strength* and *current average agent social strategy* together. Now we will define the change probability by modifying the initial definition of Equation (7) as:

$$prob_i^{l*} = \frac{\beta_1 prob_i^l + \beta_2 (1 - DL_{l,CASS} / 8)}{\sum_n (\beta_1 \cdot prob_i^n + \beta_2 \cdot (1 - DL_{n,CASS} / 8))} = \frac{\beta_1 \cdot (s_l(x, y) / \sum_{m \in L} s_m(x, y)) + \beta_2 \cdot (1 - DL_{l,CASS} / 8)}{\sum_{n \in L} (\beta_1 \cdot (s_n(x, y) / \sum_{m \in L} s_m(x, y)) + \beta_2 \cdot (1 - DL_{n,CASS} / 8))} \tag{9}$$

Where  $\beta_1$  and  $\beta_2$  are parameters to determine the relative importance of the two factors. Therefore, the probability that agent *a* selects social strategy *l* is determined by not only the impact strength of *l* but also the distance between *l* and *CASL*. Obviously, the probability in Equation (9) considers the average social strategy, which can make the agents incline to the social strategy with unification trend.

### 3.3 Concurrent Trend to Expected Average Social Strategy

Let the current social strategy of agent *i* is *l<sub>i</sub>* and its locality is (*x<sub>i</sub>*, *y<sub>i</sub>*), the expected social strategy (*ESS*) of agent *i* after the diffusion can be defined as:

$$ESS_i \approx \sum_{l \in L} prob_i^l * l \tag{10}$$

Let *n* be the number of agents in the system, then the expected average social strategy of the whole system can be defined as:

$$EASS = \left[ \sum_{i=1}^n \left( \frac{P_i}{\sum_i P_i} ESS_i \right) \right] = \left[ \sum_{i=1}^n \left( \frac{P_i}{\sum_i P_i} \sum_{l \in L} prob_i^l * l \right) \right] \tag{11}$$

With the concurrent mechanism and unification trend, each agent will incline to such *EASL*, which can make the social strategies of the system be more unified. As said above, the agent also inclines to the social strategy with strong impact strength and the *CASL*. So, we need to compromise those three inclinations together.

Therefore, the real probability that agent *i* transfers to social strategy *l* is:

$$prob_i^{l**} = \frac{\lambda_1 prob_i^l + \lambda_2 (1 - d_{l,CASS} / 8) + \lambda_3 (1 - d_{l,EASS} / 8)}{\sum_n (\lambda_1 \cdot prob_i^n + \lambda_2 \cdot (1 - d_{n,CASS} / 8) + \lambda_3 \cdot (1 - d_{n,EASS} / 8))} \tag{12}$$

$$= \frac{\lambda_1 \cdot (s_l(x, y) / \sum_{m \in L} s_m(x, y)) + \lambda_2 \cdot (1 - d_{l,CASS} / 8) + \lambda_3 \cdot (1 - d_{l,EASS} / 8)}{\sum_{n \in L} (\lambda_1 \cdot (s_n(x, y) / \sum_{m \in L} s_m(x, y)) + \lambda_2 \cdot (1 - d_{n,CASS} / 8) + \lambda_3 \cdot (1 - d_{n,EASS} / 8))}$$

Where  $\lambda_1$ ,  $\lambda_2$  and  $\lambda_3$  are parameters to determine the relative importance of the three factors. Obviously,  $\sum_l prob_i^{l**} = 1$ .

## 4 Performance Judgment and Saturation of Diffusion

### 4.1 Performance Judgment of the Diffusion

After the diffusion, there may be some social strategies remained in the system, and the social strategies can't diffuse to each other any more. Certainly, for the queue exercitation, the number of standing orientations ought to be minimized so as to make the queue orientations be unified. Otherwise, we should make the social strategy with high authority can overlay more number of agents, which can make such strategy more popular. Therefore, let  $N_L$  be the number of social strategies in the system after diffusion,  $|G(l)|$  be the agent number in the overlay group of social strategy  $l$ , then the performance criterion of the diffusion can be initially defined as:

$$P'_L = \frac{1}{N_L} \sum_{l \in L} (Rank(l) \cdot |G(l)|) \tag{13}$$

Otherwise, for the trend of unification, we will try to make the difference between every agent and the current average social strategy (CASS) after diffusion be minimized. Let the number of agents is  $n$ , the total difference between all agents and the CASL can be called as *Variation Index (VI)*:

$$VI = \frac{1}{n} \sum_{i=1}^n d_{i,CASS} \tag{14}$$

Obviously, the less  $VI$  is, the higher the performance of the diffusion is.

Therefore, we can modify the Equation (13), and define the ultimate performance criterion of the diffusion as:

$$P_L = P'_L / VI = \begin{cases} \frac{n \cdot \sum_{l \in L} (Rank(l) \cdot |G(l)|)}{N_L \cdot \sum_{i=1}^n d_{i,CASS}} & , \text{if } \sum_{i=1}^n d_{i,CASS} \neq 0 \\ \frac{n \cdot \sum_{l \in L} (Rank(l) \cdot |G(l)|)}{N_L} & , \text{else} \end{cases} \tag{15}$$

Obviously, the more  $P_L$  is, the higher the performance criterion of the diffusion is.

### 4.2 Saturation of the Diffusion

Then, how can we fix on the end of the diffusion? The end of diffusion can be called as diffusion *saturation*. In the status of diffusion saturation, the diffusion can't proceed any more. Now, the saturation of the diffusion can be defined as:

$$EASS == CASS \tag{16}$$

This means that if the expected average social strategy is equal to current average social strategy, then status of diffusion saturation is reached even though there are still changes of agent social strategies.

### 5 Case Studies and Test

Fig.4 is a case for testing our diffusion model, in the case there are 19 agents with different social strategies (standing orientations). From the top left to the bottom right, we can number the agent as  $a_1, a_2, \dots, a_{18}$ . Now we can compute the change probability of all agents respectively according to Equation (7), (9) and (12), shown in Table.1. In Table.1, the value denotes the social strategy with the highest probability.

- If we select  $prob_i^l **$ , then the diffusion can be finished within a step, and the diffusion result is that all agents have  $l_r$ .
- If we select  $prob_i^l *$ , then the diffusion result after a step is shown as Fig.5. Now we make the 2<sup>nd</sup> step diffusion for Fig.5 according to  $prob_i^l *$ , the change probability is shown as Table.2.
- If we select  $prob_i^l$ , then the diffusion result after a step is shown as Fig.6. Then, we make the 2<sup>nd</sup> step diffusion for Fig.6 according to  $prob_i^l$ . The change probabilities are shown as Table.3. Therefore, now the diffusion is finished.

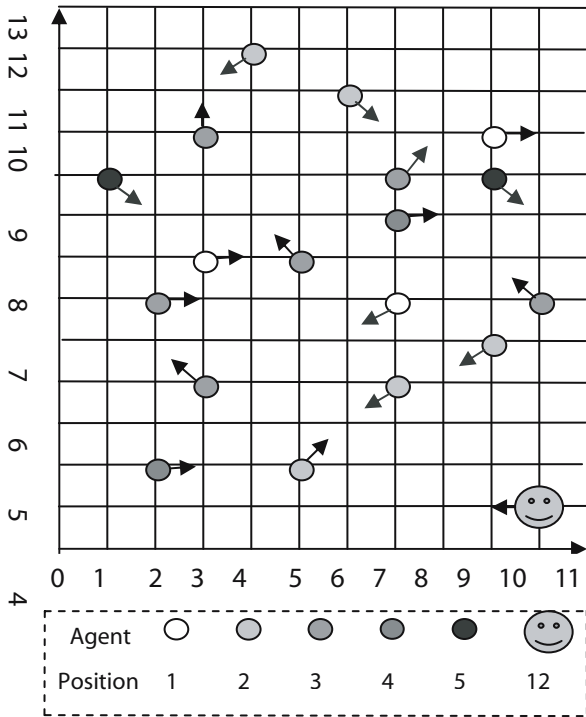
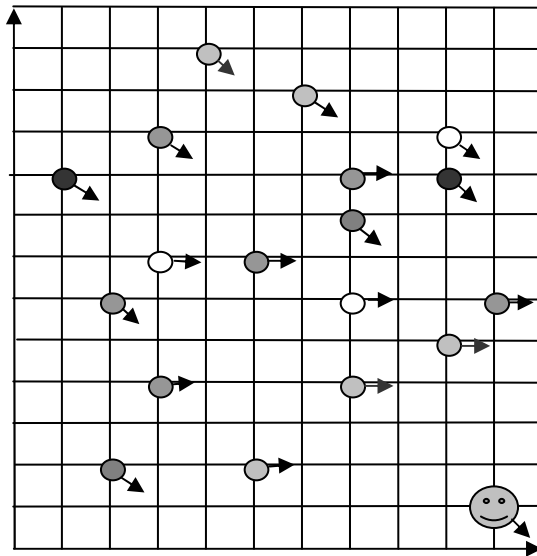


Fig. 4. The initial case, where  $P_L=12.9$ ,  $CASS=5$

**Table 1.** The inclined social strategies with the highest change probability of the initial case

Agent	$a_1$	$a_2$	$a_3$	$a_4$	$a_5$	$a_6$	$a_7$	$a_8$	$a_9$	$a_{10}$
$prob_i^l$	$l_4$	$l_3$	$l_4$	$l_4$	$l_4$	$l_3$	$l_4$	$l_3$	$l_3$	$l_3$
$prob_i^l *$	$l_4$	$l_4$	$l_4$	$l_4$	$l_4$	$l_3$	$l_4$	$l_4$	$l_3$	$l_3$
$prob_i^l **$	$l_4$	$l_4$	$l_4$	$l_4$	$l_4$	$l_4$	$l_4$	$l_4$	$l_4$	$l_4$
Agent	$a_{11}$	$a_{12}$	$a_{13}$	$a_{14}$	$a_{15}$	$a_{16}$	$a_{17}$	$a_{18}$	$a_{19}$	
$prob_i^l$	$l_3$	$l_3$	$l_6$	$l_6$	$l_3$	$l_6$	$l_3$	$l_3$	$l_3$	
$prob_i^l *$	$l_4$	$l_3$	$l_3$	$l_3$	$l_3$	$l_3$	$l_4$	$l_3$	$l_4$	
$prob_i^l **$	$l_4$	$l_4$	$l_4$	$l_4$	$l_4$	$l_4$	$l_4$	$l_4$	$l_4$	



**Fig. 5.** The agent system after the 1<sup>st</sup> diffusion by choosing  $prob_i^l *$ , now  $CASS=4, P_L=622.8$

**Table 2.** The inclined social strategies with the highest change probability  $prob_i^l *$  in Fig.5

Agent	$a_1$	$a_2$	$a_3$	$a_4$	$a_5$	$a_6$	$a_7$	$a_8$	$a_9$	$a_{10}$
$prob_i^l *$	$l_4$	$l_4$	$l_4$	$l_4$	$l_4$	$l_4$	$l_4$	$l_4$	$l_4$	$l_4$
Agent	$a_{11}$	$a_{12}$	$a_{13}$	$a_{14}$	$a_{15}$	$a_{16}$	$a_{17}$	$a_{18}$	$a_{19}$	
$prob_i^l *$	$l_4$	$l_4$	$l_4$	$l_4$	$l_4$	$l_4$	$l_4$	$l_4$	$l_4$	

Analyses for the test results:

- From the above result, we can see that it only need 1 step for the diffusion if we select  $prob_i^l **$ . When we select  $prob_i^l *$ , we consider the concurrent



**Summary:** We can also demonstrate our model in some other cases, shown in Table 4. Therefore, from the results of all cases, we can see that our model can solve the concurrent mechanism and unification trend in the agent social strategy diffusion well. For the performance criterion,  $prob_i^{l**} > prob_i^{l*} > prob_i^l$ , where ‘>’denotes that ‘performs better than’.

**Table 4.** Test results for other cases, where  $P_L^i$  denotes  $P_L$  after the  $i^{th}$  step diffusion

Case	Change Probability	Steps	$P_L^1$	$P_L^2$	$P_L^3$	$P_L^4$	$P_L^5$	$P_L^6$
1	$prob_i^l$	2	85	231				
	$prob_i^{l*}$	2	114	231				
	$prob_i^{l**}$	1	231					
2	$prob_i^l$	5	34	139	291	448	981	
	$prob_i^{l*}$	4	112	298	361	981		
	$prob_i^{l**}$	2	242	981				
3	$prob_i^l$	6	12	132	189	208	302	994
	$prob_i^{l*}$	4	34	189	245	994		
	$prob_i^{l**}$	2	213	994				
4	$prob_i^l$	3	24	109	341			
	$prob_i^{l*}$	2	98	341				
	$prob_i^{l**}$	1	341					
5	$prob_i^l$	6	12	45	176	221	304	981
	$prob_i^{l*}$	4	23	75	331	981		
	$prob_i^{l**}$	3	103	231	981			

## 6 Conclusion

There are many forms of agent social law evolution where concurrent mechanism and unification trend are often seen. This paper explores the concurrent agent social strategy diffusion with unification trend. In the concurrent diffusion, many diffusion processes will take place simultaneously, and an agent’s social strategy is not only determined by the diffusion that bears on itself but also by other diffusion processes that bear on other agents. Otherwise, with the unification trend, each agent will incline to select the average social strategy of the system. This paper provided the change probability of agent social strategy by considering the concurrent mechanism

and unification trend, which compromises the impacts of social strategy authority, concurrent mechanism, and unification trend.

## Acknowledgements

This work is supported by JSPS Research Grant No.17-05282.

## References

- [1] Y.Shoham, M.Tennenholtz. On the emergence of social conventions: Modeling, analysis and simulations. *Artificial Intelligence* 94 (1997) 139-166.
- [2] Y.Shoham, M.Tennenholtz. On social laws for artificial agent societies: off-line design. *Artificial Intelligence* 73 (1995) 231-252.
- [3] S.Onn, M.Tennenholtz. Determination of social laws for multi-agent mobilization. *Artificial Intelligence* 95 (1997) 155-167
- [4] M.ennenholtz. On stable laws and qualitative equilibria. *Artificial Intelligence* 102(1998)1-20.
- [5] Guido Boella, Leendert van der Torre. The evolution of artificial social systems. *Proceeding of the Nineteenth International Joint Conference on Artificial Intelligence*. Edinburgh, Scotland, 30 July-5 August 2005.
- [6] Yichuan Jiang, Toru Ishida. Evolve individual social laws to global social convention by hierarchical immediate diffusion. *Proceedings of the 2nd International Workshop on Massively Multi-Agent Systems*. Future University-Hakodate, Japan, 2006.
- [7] Yichuan Jiang, Toru Ishida. Collective diffusion in agent social law evolution. *Research Report*, 2006. Lab for Global Information Networks, Kyoto University.
- [8] Wolfgang Balzer, Baimo Tuomela. Collective intentions and the maintenance of social practices. *Autonomous Agents and Multi-Agent Systems*, 6, 7-33, 2003.
- [9] Taher H.Haveliwala. Topic-sensitive pagerank: a context-sensitive ranking algorithm for web search. *IEEE Transactions on Knowledge and Data Engineering*, Vol15,No.4, July/August 2003. pp.784-796.



# Exploiting Based Pre-testing in Competition Environment

Li-ming Wang<sup>1</sup> and Yang Bai<sup>2</sup>

Information Engineering School of Zhengzhou University,  
Zhengzhou 450052, China

<sup>1</sup> ielmwang@zzu.edu.cn

<sup>2</sup> bs12004@tom.com

**Abstract.** In competition environment a satisfactory multi-agent learning algorithm should, at a minimum, have rationality and convergence. Exploiter-PHC (It is written as Exploiter here) could beat many fair opponents, but it is neither rational against stationary policy nor convergent in self-play, even it could be beaten possibly by some fair opponents in lower league. Now an improved algorithm named ExploiterWT (Exploiter With Testing) based on Exploiter is proposed. The basic idea of ExploiterWT is that an additional testing period is added to estimate the Nash Equilibrium policy. ExploiterWT could satisfy these properties mentioned above. It needn't Nash Equilibrium as apriori knowledge like Exploiter when it begins to exploiting. Even ExploiterWT could avoid being beaten by some fair opponents in lower league. In this paper, at first the thoughts of this algorithm will be introduced, and then experiment results obtained in Game Pennies-Matching against other algorithms will be given.

**Keywords:** Game Theory, Nash Equilibrium, Competition, Exploiter, Reinforcement Learning.

## 1 Introduction

Agents in a MAS typically operate in large, complex, open, dynamic and unpredictable environments. The optimal policy at any moment depends on the policies of the other agents and so creates a situation of learning a moving target. Multi-agent learning is not only an intersection of Distributed Artificial Intelligence (DAI) and Machine Learning (ML), but also an area where ML and Game Theory meet. Several algorithms for multi-agent reinforcement learning have been proposed, mostly guaranteed to converge to equilibrium in the limit.

Recently Singh, Kearns and Mansour have argued the Nash Convergence of Gradient Dynamics in General-Sum Games [1]. Banerjee and Peng presented another perspective WoLF [2] on this theme and later Bowling and Veloso proposed two practical algorithms named PHC and PHC-WoLF[3],[4]. The latter could be rational and convergent to Nash Equilibrium in self-play. Chang and Kaelbling [5] have presented a new classification scheme and proposed Exploiter which could beat many fair opponents. What a pity is that Exploiter is neither convergent in self-play nor rational against stationary opponents. Banerjee and Peng pointed out the deficiency in

Kaelbling’s classification scheme. They have proposed a notion of Reactivity [6] that explains the behaviors of various algorithms. Banerjee also show that Exploiter could be beaten by some PHC-WoLF. Vincent Conitzer and Tuomas Sandholm have proposed AWESOME [7] which could be rational against stationary policy and convergent in self-play in all repeated (finite) games. In addition, many other properties have been proposed. Rob Powers and Yoav Shoham [8] proposed Targeted Optimality and Compatibility. Banerjee and Peng proposed efficient no-regret algorithm [9] and efficient algorithm of multi-step best response [10].

In this paper an algorithm named ExploiterWT(Exploiter With Testing) is proposed, it is convergent to NE in self-play and rational against stationary policy. ExploiterWT could estimate NE approximately, so it could avoid being beaten by some PHC-WoLF.

The rest of this paper is organized as follows. In Section 2, we describe some definitions about Matrix Game; In Section 3, we introduce the thoughts of PHC, PHC-WoLF and Exploiter. In Section 4, we describe the thought of ExploiterWT. In Section 5, we show the results of ExploiterWT in Matching-Pennies. Finally, in Section 6, we make some conclusions.

## 2 Definitions and Notations

Here we provide definitions of key concepts for our work.

**Definition 1.** A bimatrix game is given by a pair of matrices,  $(M_1, M_2)$ , (each of size  $|A_1| \times |A_2|$  for a two-agent game) where the payoff of the  $k$ th agent for the joint action  $(a_1, a_2)$  is given by the entry  $M_k(a_1, a_2)$ ,  $\forall (a_1, a_2) \in A_1 \times A_2, k = 1, 2$ .

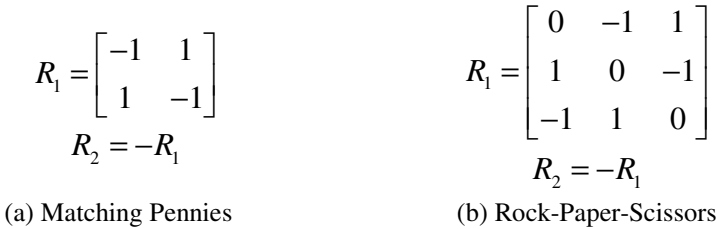


Fig. 1. Some examples of Matrix Game

A constant-sum game (also called competitive games) is a special bimatrix game where  $M_1(a_1, a_2) + M_2(a_1, a_2) = c$ ,  $\forall (a_1, a_2) \in A_1 \times A_2$ , where  $c$  is a constant. If  $c = 0$ , then it is also called a zero-sum game. Some traditional examples of single-shot matrix games are shown in Figure 1.

We consider the problem of concurrent learning in context of repeated play of a bimatrix game by two agents. A mixed policy for player  $i$ ,  $\pi_i = PD(A_i)$ , is a

probability distribution over the set of actions available to that player. The policy of the opponent of player  $i$  is written as  $\pi_{-i}$ . The expected payoff of agent  $i$  is

$$V_i(\pi_i, \pi_{-i}) = \sum_{(a_1, a_2) \in A_1 \times A_2} \pi_1(a_1) \pi_2(a_2) M_i(a_1, a_2), i = 1, 2.$$

In a repeated game, the goal of the learner (agent  $i$ ) is to deduce a policy that maximizes  $V_i(\pi_i, \pi_{-i})$  against the opponent's policy.

**Definition 2.** A best response of agent  $i$  to its opponent's policy,  $\pi_{-i}$ , is a set of probability vectors  $BR_i(\pi_{-i}) = \{\pi_i^* \in PD(A_i) \mid V_i(\pi_i^*, \pi_{-i}) \geq V_i(\pi_i, \pi_{-i}) \forall \pi_i \in PD(A_i)\}$ , where  $PD(A_i)$  is the set of probability distributions over  $A_i$ .

If both of the players are playing mutual best responses, then they are said to be in a NE (Nash Equilibrium). It is a joint policy point from where no player has any incentive for unilateral deviation, given the other's strategy. Such equilibrium is a characteristic of Matrix Game. There always exists at least one such equilibrium in mixed policies for an arbitrary finite bimatrix game [11].

A satisfactory multi-agent learning algorithm should, at a minimum, learn to play optimally against stationary opponents and converge to a Nash Equilibrium in self-play.

### 3 Discussion of Current Algorithms

Recently several algorithms have been proposed. WoLF-IGA [2] has been proven to have these two properties above in 2-player 2-action repeated games, but it assumes that the opponent's mixed policy is observable.

Several authors have applied reinforcement learning to create some algorithms. PHC (Policy Hill Climber) maintains and updates a policy based upon its history in an attempt to maximize its rewards. The basic idea of PHC is that the Q-values help us to define a gradient upon which we execute hill-climbing. PHC players play well against stationary opponents, but it could not converge in self-play. Bowling and Veloso's WoLF-PHC (Win-or-Lose-Fast-PHC) [3],[4] modifies PHC by adjusting learning rate  $\delta$  depending on whether the agent is "winning" or "losing". WoLF-PHC could satisfy rationality and convergence without the knowledge of opponent's policy.

Because Exploiter [5] models the opponent's learning algorithm rather than simply learns from his own history, it could beat many fair opponents. The idea is that Exploiter will "fool" its opponents into thinking that it is stupid by playing a decoy policy for a number of time periods and then switch to a different policy that takes advantage of their best response to its decoy policy. But Exploiter is not rational against stationary policy and not convergent in self-play.

Conitzer and Sandholm proposed AWESOME [7] that could satisfy rationality and convergence in all repeated (finite) games. The basic idea of AWESOME (Adapt When Everybody is Stationary, Otherwise Move to Equilibrium) is to try to adapt to the others' policy when they appear stationary, otherwise to retreat to a precomputed equilibrium policy. Although AWESOME is good enough, it needs another algorithm to precompute equilibrium policy.

Chang and Kaelbling [6] classified PHC and PHC-WoLF as  $H_\infty \times B_0$  learners as they have full access to their history ( $H$ ) but believe ( $B$ ) that the opponent plays a stationary policy, i.e., the opponent does not use any memory. While the Exploiter essentially belongs to the league  $H_\infty \times B_\infty$  since it believes that the opponent is playing a non-stationary policy, and learns when losing but exploits when winning by playing the maximizing action deterministically.

### 4 ExploiterWT

Exploiter is ‘cleverer’ than other learning algorithms. It could use decoy policy to “fool” its opponent and switch away from its decoy policy immediately, so it could take advantage of its opponent’s best response to its decoy policy. Unfortunately, Exploiter is not rational against stationary policy and not convergent in self-play, hence we will propose a new algorithm ExploiterWT that has these two properties. It also could exploit PHC agent and avoid being beaten by some PHC-WoLF, even it need not NE as apriori.

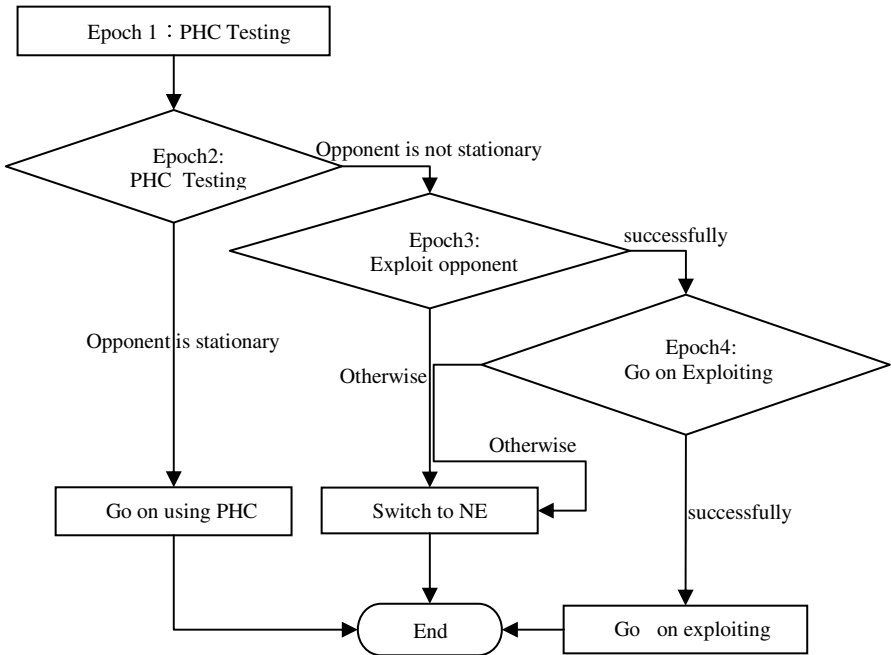


Fig. 2. The flow chart of ExploiterWT

Roughly, the basic idea of ExploiterWT is the following. We divide the game into several epochs. Firstly, ExploiterWT uses PHC to test and models his opponent at the same time. If the opponent appears to be playing stationary policy, Exploiter will use

PHC to the end of the game which will play the best response. Otherwise, ExploiterWT will exploit its opponent. Secondly, if ExploiterWT is being exploited by its opponent, it will switch to its equilibrium policy. Note that if the opponent uses WoLF, it will be convergent to equilibrium policy during epoch 1, and ExploiterWT will believe its opponent is using some stationary policy. The brief procedure is shown in Figure 2.

We will show the specification of each epoch below with an assumption that Agent 1 is playing ExploiterWT and Agent 2 other algorithm.

**Epoch 1:** In this epoch Agent 1 plays PHC to testing and calculates the average policy  $\bar{\pi}^1$  of this epoch and the average payoff  $\bar{P}^1$  of this epoch. If the opponent is playing stationary policy, Agent 1 will be convergent to its best response. If the opponent is playing PHC, the average policy of Agent 1 will be convergent to its equilibrium policy, and the average payoff of Agent 1 will be convergent to NE payoff [8]. If the opponent is playing PHC-WoLF, it will be convergent to its equilibrium policy.

**Epoch 2:** Agent 1 continues to use PHC and models its opponent, and then makes a judgment whether the opponent's policy is stationary. How can it be done? We divide epoch 2 into several time slices. In each time slice  $s$ , Agent 1 calculates the policy  $\pi_i^s$  of its opponent  $i$ .  $\Delta D_j^T$  is denoted as the max difference of the policy of agent  $j$  in epoch  $T$ , so  $\Delta D_j^T = \text{Max}_{s,t \in T} |\pi_i^s - \pi_i^t|$ , where  $i = 1, \dots, |A_j|$ . We conclude that the opponent is playing stationary policy if and only if  $\Delta D_j^T < \varepsilon_s$ , where  $\varepsilon_s > 0$ .

Even if the policy of the opponent is actually stationary, there is a fixed nonzero probability that the observed actions taken in any given epoch, by chance, do not appear to be drawn from the stationary policy. We can increase  $\varepsilon_s$  and increase the length of each time slice in order to eliminate this possibility. We can deduce  $\lim_{n \rightarrow \infty} P(|\pi_i^s - \pi_i^*| < \varepsilon) = 1$  by Bernoulli theorem, where  $\pi_i^*$  is denoted the real policy of Agent  $i$  in time slice  $s$  and  $n$  is the size of  $s$ . If  $\varepsilon_s = 2\varepsilon$ , then we can get  $\lim_{n \rightarrow \infty} P(|\pi_i^s - \pi_i^*| < \varepsilon_s) = 1$ .

If the policy of the opponent is stationary, Agent 1 will keep using PHC to the end. It will be convergent to BR. If the opponent is PHC-WoLF, it has been convergent to its equilibrium policy in epoch 1. Agent 1 will conclude that its opponent is playing stationary policy in this epoch, and do the same thing as against stationary policy. If the policy of the opponent is not stationary, Agent 1 will use Exploiter in the next epoch.

**Epoch 3:** Agent 1 plays Exploiter and calculates the average payoff  $\bar{P}^3$  of this epoch. Exploiter need NE as apriori obtained in epoch 1, i.e.  $\bar{\pi}^1$ . In this epoch, when Agent 1 exploits its opponent successfully, if the opponent plays ExploiterWT too, the joint policy would come to a halt[5] at which one of the player would win, and the other would lose, i.e., ExploiterWT may exploit its opponent or be exploited by its opponent. Agent 1 will switch to its equilibrium policy when it could not exploit its opponent successfully.

How could we judge whether Agent 1 is winner? We know that we have got NE payoff in epoch 1, i.e.,  $\bar{P}^1$ , so if  $\bar{P}^3 - \bar{P}^1 > \epsilon_p$ , where  $\epsilon_p > 0$ , we can conclude that Agent 1 is winner. The principle of the method is as the same as upon.

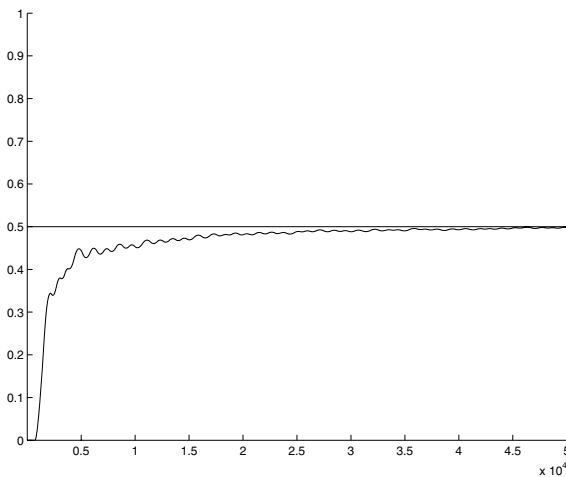
**Epoch 4:** Agent 1 continues to use Exploiter and calculates the average payoff  $\bar{P}^4$  at the end of this epoch. If  $\bar{P}^4 - \bar{P}^1 > \epsilon_p$ , Agent 1 is winner in this epoch, then it keeps using Exploiter to the end. Otherwise if Agent 1 could not exploit its opponent successfully, it would switch to its equilibrium policy to the end. The latter case would happen in self-play and Agent 1 wins in epoch 3. When Agent 1 could not gain more rewards, it will switch to its equilibrium policy. Thus, both of the players reach their NE policy.

To sum up, epoch 1 make WoLF convergence. We judge whether the policy of the opponent is stationary in epoch 2. We judge the policy of opponent is PHC or ExploiterWT in the later two epochs.

### 5 Experiments and Analysis

We will use the ExploiterWT algorithm which is described above to play against several algorithms in repeated game Matching Pennies. The NE of this game is (0.5, 0.5) and corresponding average payoff is (0, 0).

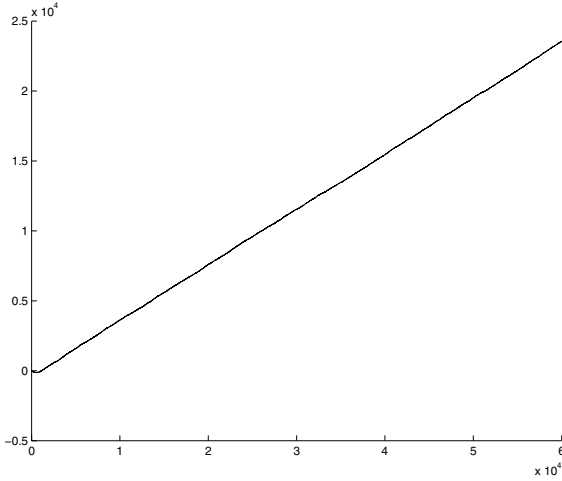
The key of ExploiterWT is testing period in which we could estimate the NE policy by calculating the average policy if the opponent is using PHC i.e. the average policy of agent using PHC is convergent to NE in self-play. Figure 3 shows the PHC’s average policy trajectory against PHC. We can see that the average policy is convergent to the NE policy indeed, and the more the testing period is long, the more the estimated NE is accuracy.



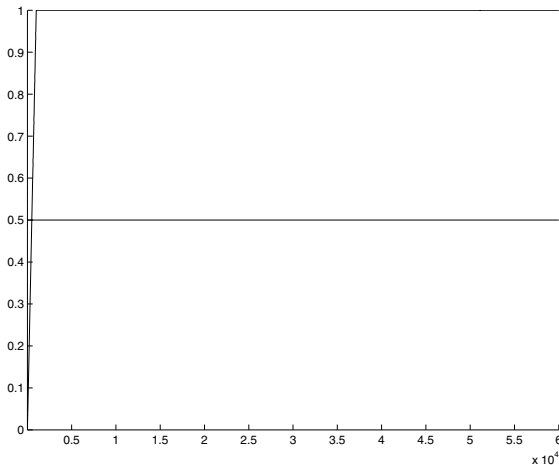
**Fig. 3.** The PHC’s average policy trajectory against PHC

Now we show the results of the ExploiterWT playing against other algorithms. Let Agent 1 plays ExploiterWT and Agent 2 plays other algorithms in turn. Let the length of epoch 1 is 20000, and the length of epoch 2, 3, 4 is 10000 each. After these epochs the game will continue playing 10000 times.

Figure 4 shows the cumulative gains of Agent 1 when Agent 2 plays stationary policy ( $\pi_2 = 0.3$ ). In this case, Agent 2 does not use its Nash equilibrium policy, so Agent 1 could gain much more rewards. We can see Agent 1 does not change its algorithm PHC after epoch 2, so ExploiterWT could be convergent to its BR against stationary policy. Figure 5 shows the ExploiterWT's policy trajectory. We can see that Agent 1 is convergent to its BR ( $\pi_1 = 1$ ) indeed.

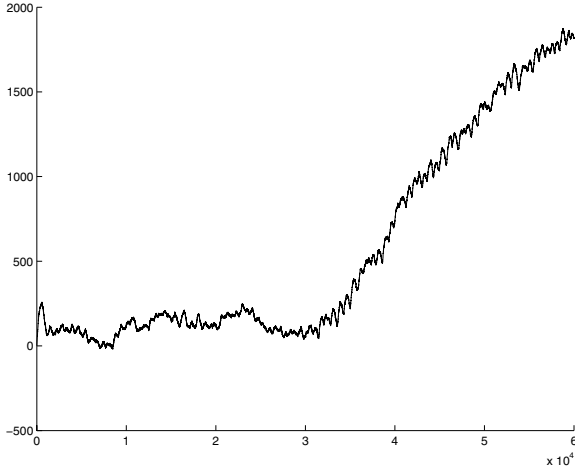


**Fig. 4.** The cumulative gains of ExploiterWT against stationary policy ( $\pi_2 = 0.3$ )

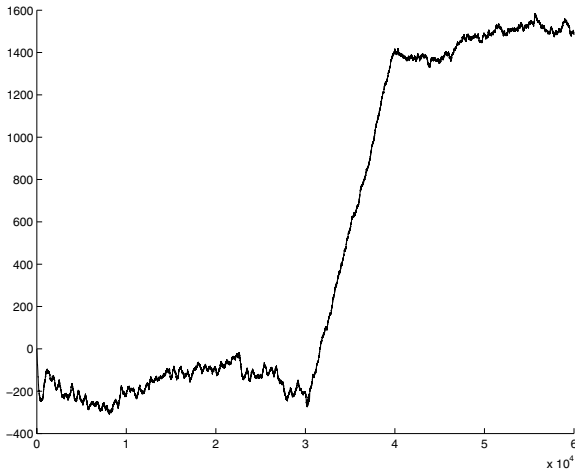


**Fig. 5.** The ExploiterWT's policy trajectory against stationary policy ( $\pi_2 = 0.3$ )

Figure 6 shows the cumulative gains of Agent 1 when Agent 2 plays PHC. During the first 30000 times plays Agent 1 conclude that its opponent is not stationary, and at the same time Agent 1 could estimate its NE policy which will be used in Exploiter, then it begins to exploit its opponent. Because Agent 1 could get more rewards during epoch 3, it will continue exploiting. At the 40000<sup>th</sup> play Agent 1 judge it is still a winner, so it will keep Exploiter to the end.



**Fig. 6.** The cumulative gains of ExploiterWT against PHC



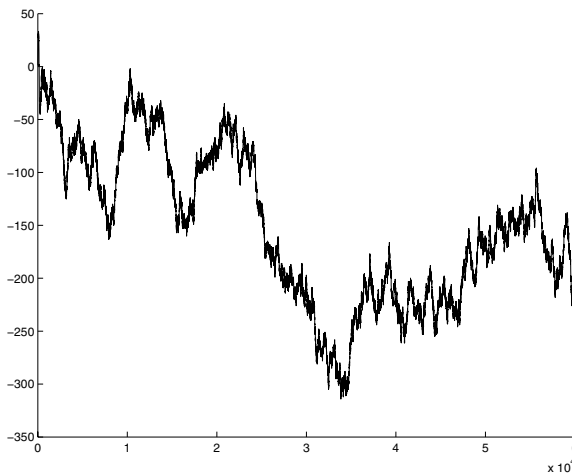
**Fig. 7.** The cumulative gains of ExploiterWT in self-play

Figure 7 shows the cumulative gains of Agent 1 when Agent 2 plays ExploiterWT. Both of the Agents play PHC in epoch 1 and 2. They both estimate the equilibrium policy themselves, and conclude that the opponent is not playing stationary policy, so



both of them will play Exploiter afterwards. In Figure 7, we see that Agent 1 get more rewards than Agent 1 during epoch 3, so Agent 2 switches to its equilibrium policy at the end of epoch 3. Because Agent 1 finds it could not get more rewards in epoch 4, it switches to its equilibrium at the end of epoch 4. Finally, they both play their NE policy themselves.

Figure 8 shows the cumulative gains of Agent 1 when Agent 2 plays WoLF. The accumulated payoff of Agent 1 is nearly 0 during the whole game. The gains decreased slightly in the first 30000 plays. It is due that the 'Reactivity' [6] of PHC-WoLF is higher than that of PHC. After the 20000<sup>th</sup> play, Agent 2 convergences to its equilibrium, so Agent 1 believe its opponent is playing stationary policy, Agent 1 remains PHC to the end. We can see that during epoch 3, 4 the amplitude is smaller than that before during epoch 1, 2. It is important that Agent 1 must select an appropriate decay function for learning rate in this case, just like PHC-WoLF in self-play [3].



**Fig. 8.** The cumulative gains of ExploiterWT against WoLF

## 6 Conclusions

In this paper, The ExploiterWT improved from Exploiter is proposed, which could estimate its equilibrium approximately by a testing process. ExploiterWT also could estimate its opponent's policy by modeling its opponent. ExploiterWT is rational against stationary policy and convergent in self-play. It could still exploit PHC agent. ExploiterWT need not a NE as apriori, even it could avoid being beaten by WoLF. Because it could not estimate its equilibrium, ExploiterWT could not convergence to NE against pure Exploiter. Several experiments proved ExploiterWT could satisfy these properties above. In the future, we would like to extend our framework to more general stochastic games with multiple states and multiple players.

## References

1. S.Singh, M.Kearns, Y.Mansour, Nash Convergence of Gradient Dynamics in General-Sum Games, in: Proceedings of the Sixteenth Conference on Uncertainty in Artificial Intelligence, Morgan Kaufman, 2000, pp.541-548.
2. Bikramjit Banerjee and Jing Peng, Convergent Gradient Ascent in General-Sum Games, ECML 2002:1-9
3. M.Bowling and M.Veloso. Multi-agent learning using a variable learning rate. Artificial Intelligence, 136:215-250,2002.
4. M.Bowling and M.Veloso. Rational and convergent learning in stochastic games. In proceedings of the Seventeenth International Joint Conference on Artificial Intelligence, 2001.
5. Y.-H. Chang and L.P. Kaelbling. Playing is believing: The role of beliefs in multi-agent learning. In Neural Information Processing Systems, Vancouver, Canada, 2001
6. Bikramjit Banerjee, and Jing Peng, The role of reactivity in multi-agent learning, in the Proceedings of the Third International Joint Conference on Autonomous Agents and Multi-agent Systems, Jul19-23, 2004 in New York,NY
7. Vincent Conitzer and Tuomas Sandholm. AWESOME: A General Multi-agent Learning Algorithm that Converges in Self-Play and Learns a Best Response Against Stationary Opponents. In Proceedings of the 20th International Conference on Machine Learning (ICML-03), pp. 83-90, Washington, DC, USA, 2003.
8. Rob Powers, Yoav Shoham, New Criteria and a New Algorithm for Learning in Multi-Agent Systems, NIPS 2004
9. Bikramjit Banerjee and Jing Peng, Efficient No-regret Multiagent Learning, in the Proceedings of the Twentieth National Conference on Artificial Intelligence, Jul 9 - 13, 2005 in Pittsburgh, PA. (AAAI-05)
10. Bikramjit Banerjee and Jing Peng, "Efficient Learning of Multi-step Best Response", in the Proceedings of the Fourth International Joint Conference on Autonomous Agents and Multiagent Systems, Jul 25 - 29, 2005 in Utrecht. (AAMAS-05)
11. J.F Nash. Non-cooperative games. Annals of Mathematics, 54:286-295, 1951.

# Teamwork Formation for *Keepaway* in Robotics Soccer (Reinforcement Learning Approach)

Nobuyuki Tanaka and Sachiyo Arai

Chiba University, 1-33 Yayoi-cho, Inage, Chiba 263-8522, Japan  
n.tanaka@graduate.chiba-u.jp, sachiyo@faculty.chiba-u.jp

**Abstract.** In this paper, we discuss guidelines for a reward design problem that defines when and what amount of reward should be given to the agents, within the context of reinforcement learning approach. We take keepaway soccer as a standard task of multiagent domain which requires skilled teamwork. The difficulties of designing reward for good teamwork are due to its features as follows: i) since it is a continuing task which has no explicit goal, it is hard to tell when reward should be given to the agents, ii) since it is a multiagent cooperative task, it is hard to make a fair share of the reward for each agent's contribution. Through some experiments, we show that reward design have a major effect on the agent's behavior, and introduce the reward function that makes agents perform keepaway successfully.

## 1 Introduction

We are concerned with designing a good reward measure for a continuing task of multiagent domains. We have been interested in *keepaway* soccer [1,2,3] where a team tries to maintain a ball possession ducking away from the opponent's interruptions. *Keepaway* soccer problem, originally suggested by Stone[1], provides a basis for discussing various issues of multiagent systems and reinforcement learning problem[4]. So far, to our knowledge, *designing a reward function problem* had been left outside of reinforcement learning research yet, and reward function introduced by Stone[3] has been commonly used. However, this problem became to be discussed as an important issue recently[5]. Difficulties of a designing reward measure for *keepaway* are mainly due to its features as follows: First, it belongs to a *continuing task* that has no explicit goal to achieve. Second, it is a *multiagent cooperative task* where there exists a reward assignment problem to emerge desirable teamwork.

As for the aspect of *continuing task*, we can consult the case of single-agent continuing tasks such as a *pole balancing task*, where one episode consists of a period from starting state to failure state. In such a task which will end up with failure, penalty could be a good reward measure to evaluate teamwork and individual skills as well. Meanwhile, as for the aspect of *multiagent task* including both teammate and opponent, it is hard to tell who contributes to the task. Here, it should be noted that increasing each keeper's cycle will not always

lead to increase the total cycles of team's. That is, it is not always suitable to assign positive reward to each agent according to the amount of time cycles of each. If we can assign an individual reward of each agent appropriately, it will more effective on cooperation than sharing the common reward within the team. But if it is not appropriate, it will lead to worse performance than sharing the common reward. Therefore, assigning individual reward to each agent seems something of a double-edged sword. Thus, we would like to focus on assigning reward measure so as not to affect harmfully on multiagent learning.

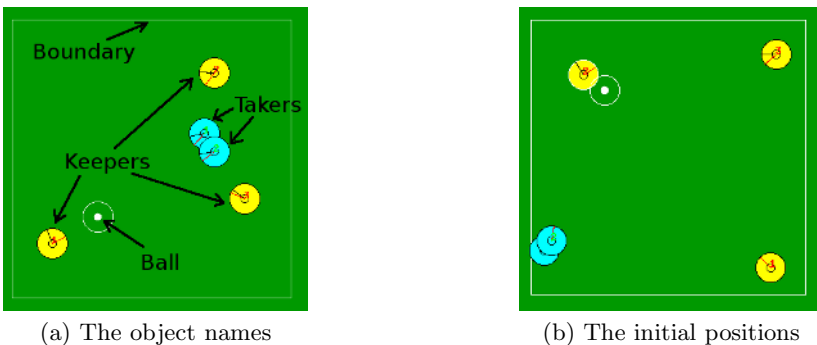
The following part of this paper is organized as follows. In the next section, we describe the formalization of *keepaway* soccer domain, and discuss its features from the viewpoint of reinforcement learning. In Section 3, we introduce the reinforcement learning algorithm we applied, and our reward design for *keepaway* in consideration for the features of it. Section 4 shows our result and acquired behavior of agents', then we discuss how far our reward design can be available on the reinforcement learning tasks in Section 5. Lastly, we would like to state our conclusion and future work in Section 6.

## 2 Problem Domain

### 2.1 Keepaway Soccer

*Keepaway*[1] is known as the subtask of RoboCup soccer, and it provides the great basis for discussion on important issues of multiagent systems. It consists of *keepers* who try to keep a ball possession, and *takers* who attempt to take possession of the ball within a limited region. The *episode* terminates whenever takers take possession or ball runs off the region, and players are reset for the new episode. As for takers, when takers keep the ball more than four cycle of simulation time, takers are deemed to get the ball possession successfully.

Figure 1 shows the case of three keepers and two takers (calls 3vs.2) playing in the 20[m]×20[m] sized region. Here,  $K_1$  is the keeper currently having the ball,  $K_2$  is the closest one to  $K_1$ ,  $K_3$  is the next closest one, and so on up to



**Fig. 1.** *Keepaway*: 3vs.2 Keepaway Task in 20[m]× 20[m]

$K_n$  when there exist  $n$  keepers in the region. In a similar way,  $T_1$  indicates the closest taker to  $K_1$ ,  $T_2$  is the next closest one, and so on up to  $T_m$  when there exist  $m$  takers in the region.

## 2.2 Macro-actions

In RoboCup soccer simulation, each player executes a primitive action such as  $\text{turn}(\text{angle})$ ,  $\text{dash}(\text{power})$  or  $\text{kick}(\text{power}, \text{angle})$  every 100[ms]. However, it is difficult to employ these primitive actions when we take reinforcement learning approach to this domain, because the parameters of actions and state variables have continuous values that make state space very huge and complicate. To avoid the state representation problem, macro-actions which were proposed by Stone<sup>1</sup> is very helpful, and we employ these macro-actions as follows.

**HoldBall():** Remain stationary while keeping possession of the ball in a position that is as far away from the opponents as possible.

**PassBall( $k$ ):** Kick the ball directly towards keeper  $k$ .

**GetOpen():** Move to a position that is free from opponents and open for a pass from the ball's current position.

**GoToBall():** Intercept a moving ball or move directly towards a stationary ball.

**BlockPass( $k$ ):** Move to a position between the keeper with the ball and keeper  $k$ .

Since each macro-action consists of some primitive actions, it requires more than one step [100ms/step]. Therefore, the *keepaway* task should be modeled as a *semi-Markov* decision process(SMDP). In addition, in the case of RoboCup soccer simulation, it assumes that there exist noises on the visual information during *keepaway* task. Considering the above features of task model, a distance to an object from player is defined as the following equation.

$$d' = \text{Quantize}(\exp(\text{Quantize}(\log(d), q), 0.1)) \quad (1)$$

$$\text{Quantize}(V, Q) = \text{rint}(V/Q)Q \quad (2)$$

Here,  $d'$  and  $d$  are quantized value and exact value of the distance respectively. The function  $\text{rint}(x)$  truncates a number  $x$  of decimal places. The parameter  $q$  is set as 0.1 and 0.01 when an object is moving and fixed respectively. The noise parameter ranges from 1.0 to 10.0. For example, when distance between players is less than 10.0[m], noise parameter is set by 1.0, and when distance is 100.0[m], the most noisy case, it will set by 10.0.

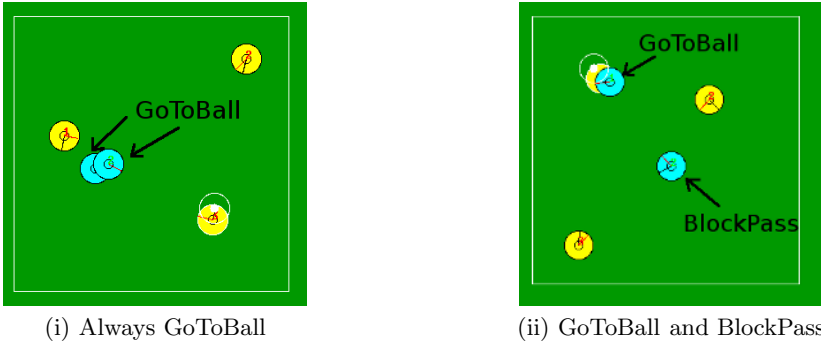
## 2.3 Takers' Policy

In this paper, we consider two types of takers' policy, as shown in Figure 2, to see the effects of the reward design. In the case of policy-(i), takers select

<sup>1</sup> Learning To Play Keepaway:

<http://www.cs.utexas.edu/users/AustinVilla/sim/keepaway/>

GoToBall() to interrupt keeper's HoldBall() whenever either taker is in a certain distance from the ball. Whereas, in the case of policy-(ii), taker who is in the nearest position to the ball selects GoToBall(), and the other taker selects BlockPass( $k$ ). The policy-(ii) is more strategic than policy-(i) because each taker plays a distinct role in the policy-(ii) to intercept a pass.



**Fig. 2.** Takers' policy: 3vs.2 Keepaway Task in a 20[m]  $\times$  20[m] region

## 2.4 Issues of Reward Design

As we mentioned in the previous section, difficulties of designing reward are due to the lack of an explicit goal, and the number of agents that involved with task. *Keepaway* task contrasts with *pursuit game*, a traditional multiagent research testbed, that has the explicit common goal. Because *pursuit game* is an episodic task, the reward just has to be given when hunters achieve the goal. Besides, it is easier to assign hunter(learner)'s reward than the reward of keeper(learner)'s, because all four hunters definitely contributed to capture the prey. Therefore, *keepaway* is classified into harder task in terms of designing reward because we have no clues to define an explicit goal beforehand, as well as to assign reward to each agent.

From the aspect of *continuing task*, we can consult the case of single-agent continuing tasks, e.g., a *pole balancing*, one episode consists of a period from starting state to the failure state. In such a continuing task, an episode will end up with failure, and penalty can provide an indication of designing good reward measure to improve teamwork and individual skills as well. Meanwhile, from the aspect of *multiagent task* including both teammate and opponent, it is hard to tell who did contribute to keeping a ball possession within the team. In other words, we should consider the case where some keepers may contribute and others may not, or an opponent(taker) may assist to keep a ball as it happens. What has to be noticed is that the episode of multiagent's continuing task will be ended by someone's failure. We call the issue of deciding when and what amount of reward should be given to the agents, a *reward design* problem.

### 3 Approach

#### 3.1 Learning Algorithm

We use Sarsa( $\lambda$ ) with replacing eligibility trace[6] for the learning algorithm of keeper's by following Stone's approach[3]. With regard to the state representation, we use a tile-coding[7] as function approximations for continuous state variables. Figure 3 shows a learning algorithm that we use. In line 4 and 8, feature vector  $\mathcal{F}_t$  and  $\mathcal{F}_{t+1}$  are made up by tile-coding. In our experiments, we used the primarily single-dimensional tilings. 32 tilings are overlaid with 1/32 offset, and each tiling is divided into  $n_{tile}$  segments. Each segment is called a *tile* and each state variable lies in a certain tile which is called an *active tile*.

In *keepaway*, distances between players and angles are represented as state variables. In the case of 3vs.2, the total number of state variables is 13 that consists of 11 for distance variable and 2 for angle variable. In our experiments,  $n_{tile} = 10$  for each 11 distance variables,  $n_{tile} = 18$  for each 2 angle variables.

Accordingly, the number of total tiles  $N_{tile}$  is 4672. Each value of the state variable  $i$  is represented as a feature vector,  $\mathcal{F}(i)$ , and  $\sum_{i=1}^{N_{tile}} \mathcal{F}(i) = 1$ . Each value of  $i$  is shown as follows:

$$i = \begin{cases} \frac{n_{tile}}{N_{tile}} & (\text{ith tile is active}) \\ 0 & (\text{otherwise}) \end{cases} \quad (3)$$

```

initialize  $\theta(i, a), e(i, a)$ 
1  each episode:
2    each SMDP step:
3      keeper gets  $s_t$  from environment
4      make up a feature vector  $\mathcal{F}_t(i)$  from  $s_t$ 
5       $Q_a = \sum_{i=0}^{N-1} \theta(i, a)\mathcal{F}_t(i)$  for all  $a$ 
6      select action  $a_t$  using  $Q_a$ 
7      gets  $s_{t+1}, r_{t+1}$ 
8      make up a feature vector  $\mathcal{F}_{t+1}(i)$  from  $s_{t+1}$ 
9       $Q_a = \sum_{i=0}^{N-1} \theta(i, a)\mathcal{F}_{t+1}(i)$  for all  $a$ 
10     select action  $a_{t+1}$  using  $Q_a$ 
11     for all  $i$ 
12        $\delta = r_{t+1} + \gamma Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t)$ 
13        $\theta(i, a) = \theta(i, a)e(i, a)\delta$ , for all  $a$ 
14        $e(i, a) = \lambda e(i, a)$ , for all  $a$ 
15        $e(i, a) = \mathcal{F}_{t+1}(i)$ , if  $a = a_{t+1}$ 
16        $e(i, a) = 0$ , if  $a \neq a_{t+1}$ 
17      $s_t \leftarrow s_{t+1}$ 
18   if episode ends, go to line 2

```

**Fig. 3.** Learning Algorithm: Sarsa( $\lambda$ ) with replacing eligibility trace (Satinder P. Singh, 1996)

### 3.2 Previous Work in Reward Design

As for the reward  $r$  appeared in line 7 of Figure 3,  $r_s$  defined by Eq.4[3] is commonly used for *keepaway*. Here, *CurrentTime* is the simulation time when keeper holds the ball or episode ends, and *LastActionTime* is the simulation time when keeper selects the last action. We refer to function Eq.4 as  $r_s$  afterward. In this approach, reward is defined as the amount of time period between the last action and the current time (or end time). That is, the longer amount of time has passed after taking off the ball, the higher amount of reward will be given to the keeper.

$$r_s = \text{CurrentTime} - \text{LastActionTime} \tag{4}$$

This setting seems reasonable and proper way, seemingly. However, there are some problematic cases by using  $r_s$  as shown in Figure 4 and Figure 5, for example. In Figure 4(b),  $K_1$ , who is a current ball holder, gets larger reward than one in the case of Figure 4(a). Consequently, keeper becomes to pass the ball to the intercepting takers on purpose, because the ball will bounce back directly to  $K_1$ , then  $K_1$  is paid some reward for selecting this action. This action seems to yield a larger reward than other actions.

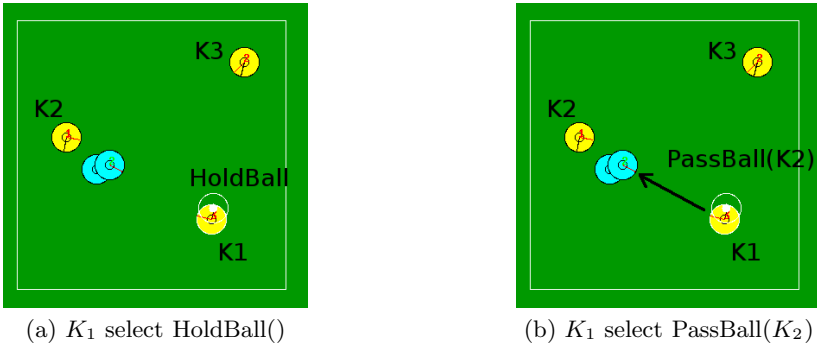


Fig. 4. Problematic situation 1

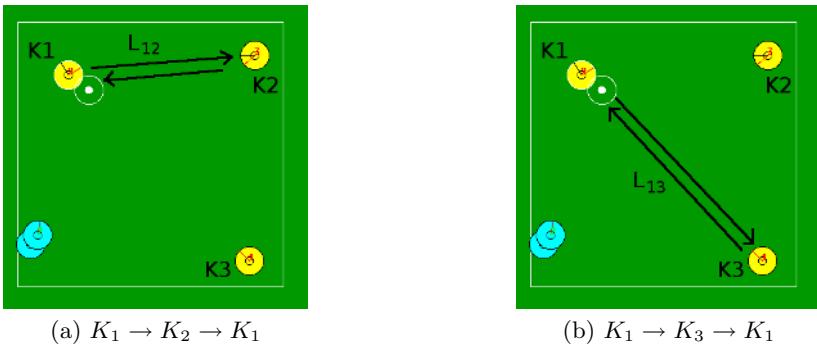


Fig. 5. Problematic situation 2



In the case of Figure 5,  $K_1$  in Figure 5(b) gets larger reward than one in Figure 5(a) when reward is defined by  $r_s$ . Consequently, keeper likely to pass the teammate(keeper) who is in the farthest position to get larger reward than pass to the nearer one. Although it seems reasonable, we can't tell which one is the better strategy because it depends on the amount of noise and keeper's skill.

These examples show the difficulty of designing a reward function for the continuing task, as we mentioned previously.

### 3.3 Our Approach for Reward Design

In the well-known continuing task, *pole balancing*, the amount of reward is defined by Eq. 5. The agent will get 0 and  $-1$  when it is in successful and failure condition, respectively. We follow the manner that the reward should make agent do what it to achieve, not how it achieves [7]. From this standpoint, the reward value has to be constant during successful situation ,such as *pole balancing* because we do not know which action achieves the task. While  $r_s$  by Eq.4[3] provides differentially programmed reward to each step as shown in Figure 6. Usually, it is hard to say whether these values become appropriate indicator to keep successful situation or not. Therefore, we introduce the novel reward function based on a constant reward sequence (Figure 6) to inhibit harmful effects of the reward design on emerging behavior.

$$r = \begin{cases} -1 & (\text{under a failure condition}) \\ 0 & (\text{otherwise}) \end{cases} \quad (5)$$

The major difference between the domain of *pole balancing* and *keepaway* is the number of agents involved. Unlike in the single-agent case where one agent is responsible for failure/success of task, in the multiagent case, responsibility is diffused. However, in *keepaway* task, specifying who causes failure seems much

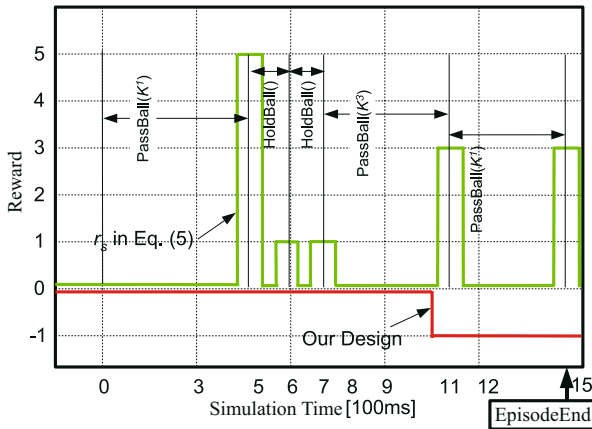


Fig. 6. Reward Sequence in Continuing Task

easier than specifying who contributes success. Therefore, we design reward functions as shown in Eq.6 where  $T$  is given by  $EpisodeEndTime - LastActionTime$  when task fails. Otherwise, agent receives 0 constantly during successful situation. The reason why sharing the same reward ( $= 0$ ) among the agents in the successful situation is that we could not tell which agent contributes the task from the length of keeping time. Though we could know one agent keeps task longer locally, it does not always causes good teamwork.

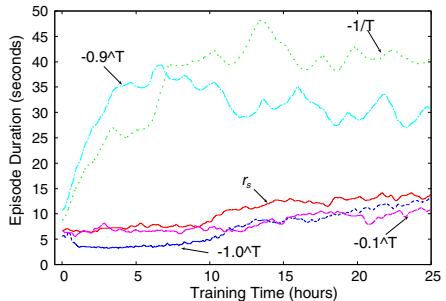
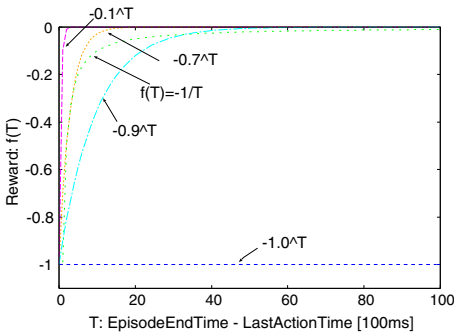
In our design, each agent will receive the reward  $f(T)$  according to its amount of time period: i.e., from taking off the ball ( $LastActionTime$ ) to the end of task( $EpisodeEndTime$ ), at the end of each episode. We would make function  $f(T)$  fulfill  $f(T) \leq 0$ , and reflect the degree of success for agents' joint action sequences. Figure 7 shows some examples of function  $f(T)$  by which keeper who terminated the episode receives the largest penalty (i.e., the smallest reward). Here, x and y-axis indicate length of  $T$  defined by Eq.6 and value of  $f(T)$ , respectively. In our experiments, we use  $f(T) = -\beta^T$  and  $f(T) = -1/T$  as reward functions. For simplicity, we call  $f(T) = -1/T$ ,  $r_f$  in the following sections.

$$r(T) = \begin{cases} f(T) & (\text{under failure condition}) \\ 0 & (\text{otherwise}) \end{cases} \tag{6}$$

$$T = EpisodeEndTime - LastActionTime$$

### 4 Experiment

In this section, we show the empirical results in the *keepaway* domain. The learning algorithm is shown in Figure 3, and the parameters are set as  $\alpha = 0.125$ ,  $\gamma = 0.95$ ,  $\lambda = 0$ , and  $\epsilon = 0.01$  for  $\epsilon$ -greedy. As for the noise parameter  $q$  as mentioned in Section 2.2, we set  $q$  by 0.00001, the same setting of Stone's[3], to see the behavior of noise-free environment.



**Fig. 7.** Reward function under different  $\beta$  **Fig. 8.** Learning curve under the various reward functions(100 episodes moving average): against Takers' policy (i)

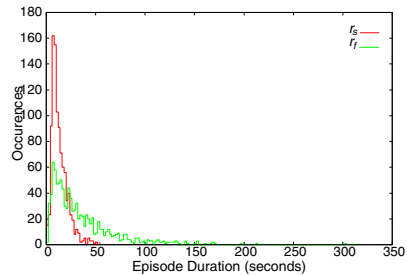
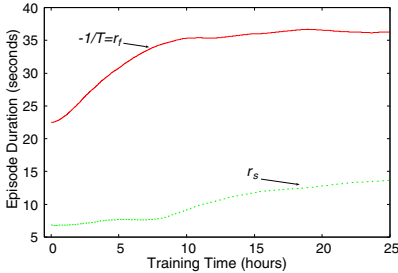
Figure 8 shows learning curves of five different reward functions that introduced in Figure 7. Here, x and y-axis indicate length of training time, and the length of keeping time, respectively. We plot the moving average of 100 episodes.

### 4.1 Performances

In the case of  $f(T) = -1$ , the same reward function of a *pole balancing* problem, the performance declines in early learning stage as shown in Figure 8. The major reason of decline is due to the multiagent learning problems, such as *reward assignment* and simultaneous learning.

Giving  $-1$  to all keepers under failure conditions, both keepers who select appropriate and inappropriate actions equally receive the penalty  $-1$ . This causes harmful effect on learning, especially in the initial stage. However, after considerable learning, all keepers learned better action. We can find that the cases of  $\beta = 0.1$  and  $\beta = 1.0$  draw similar curves in Figure 8 though each value of  $f(T)$  (Figure 7) are totally different. The likely explanation for the similarity is due to the domain of  $f(T)$ . In  $T > 1$ , the case of  $\beta = 0.1$  reaches 0, and the case of  $\beta = 1.0$  reaches 1 respectively as shown in Figure 7. This indicates that all keepers will receive the same reward value, and reward assignment problem causes the harmful effects in both cases.

Figure 9 shows the learning curves to compare two cases with different reward functions. One is  $r_s$  and the other is  $r_f = -1/T$ . Because  $r_f$  shows the best performance of keepers', we focus on this reward function, which is based on the time of failure. Figure 10 shows histograms of episode duration of keepers' who experienced 25 hours keepaway training. In Figure 10, keepers who learned by  $r_s$  possess the ball for up to about 100 seconds, while keepers who learned by  $r_f$  possess the ball for more than 300 seconds.



**Fig. 9.** Learning curves with the best-performance function and the existing function(1000 moving average):against Takers' policy (i)

**Fig. 10.** Histograms after 25 hours learning,  $0 \sim 180$  episode duration

### 4.2 Emerging Behavior

**Effect of Reward Design.** We compare the acquired behaviors of both cases,  $r_s$  and  $r_f$  after 25 hours training. One of the notable differences between the two

cases is the timing of the pass. Figure 11(a) and (b) show the behaviors of  $K_1$  reinforced by  $r_s$  and  $r_f$  respectively. Keeper with  $r_f$  shown in (b) does not pass the ball until the takers become quite near to him. While, keeper with  $r_s$  seems to pass regardless of takers' location. To examine the difference of pass timing between (a) and (b), we compare the distance from the keeper having the ball to the nearest taker of each reward function. The distance in the case of using  $r_f$  is about 2[m] shorter than one in the case of  $r_s$  as shown in Table 1. The behavior (i.e., keeper holds the ball until takers becomes quite near to a keeper) means that keepers often select HoldBall(). Then, we focused the change of pass frequency within 1 seconds in the halfway of learning process as shown in Figure 12, and Table 2 shows the pass frequency after learning. Here, we found that the frequency of passing a ball was smaller in the case of  $r_f$  than that of  $r_s$ .

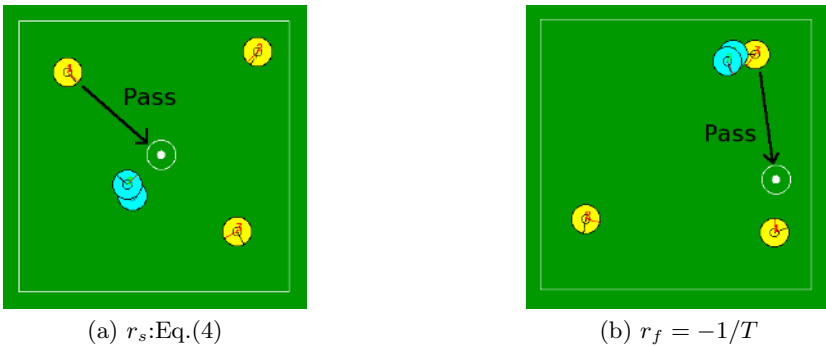


Fig. 11. Pass Timing

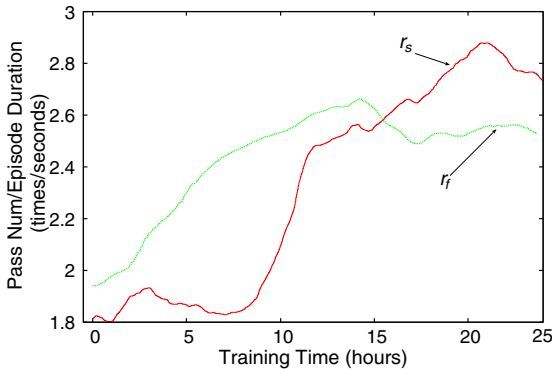
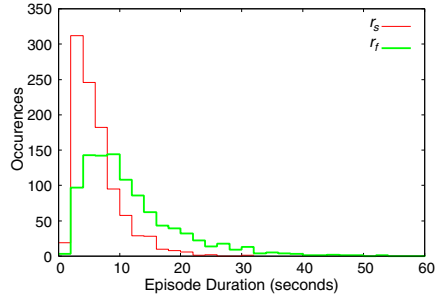
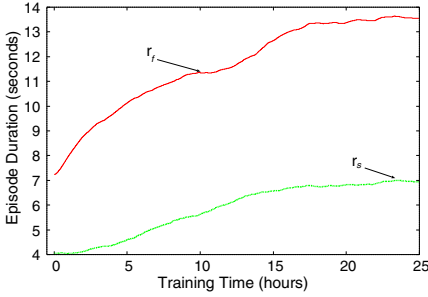


Fig. 12. Pass frequency in halfway of learning

**Effect of Takers' Policy.** Here, we discuss the emerged behavior from the view point of takers' policies introduced in Figure 2(i) and (ii). It seems that emerged behavior, shown in Figure 11(a), is especially effective against the takers with



**Fig. 13.** Learning curves with the best reward function and the existing function, 0 ~ 350 episode duration (1000 moving average): against Takers' policy (ii)

**Table 1.** Pass timing: distance to the nearest taker from keeper with the ball

	distance [m]	
	takers' policy-(i)	takers' policy-(ii)
$r_f$	5.60	3.50
$r_s$	7.44	7.38

**Table 2.** Pass Frequency: (the number of pass during 1 second)

	frequency [times/seconds]	
	takers' policy-(i)	takers' policy-(ii)
$r_f$	0.425	1.293
$r_s$	1.344	1.550

policy-(i). Because both takers always select `GoToBall()`, keeper  $K_2$  and  $K_3$  are always free from takers. Thus, we should examine the availability of our reward function,  $r_f$ , in the different situation where takers have more complicated policy than policy-(i). Then, we apply our reward function to the situation where takers act with policy-(ii). Figure 13 shows the comparison between two learning curves with our keepers using  $r_f$  and  $r_s$ .

In the case where takers act with policy-(ii), the performance of keepers with  $r_s$  gets worse than that of policy-(i). As mentioned in 3.2, reward function  $r_f$  have adverse effects on learning. Also in the case of takers using policy-(ii),  $r_f$  makes keepers possess the ball longer than function  $r_s$ . Figure 14 shows histograms of episode duration after keepers experienced 25 hours training against takers who act with policy-(ii). We found that keepers learned by  $r_f$  could possess the ball twice as longer duration as the ones learned by  $r_s$ . However, episode duration becomes shorter than that against takers with policy-(i) as shown in Figure 2(i), because of sophisticated takers' policy. While, there is not large difference of pass frequencies between the cases of  $r_f$  and  $r_s$ , because takers play the distinct roles by using policy-(ii) where one of takers can immediately reach the keeper who receives the ball, and keeper with the ball must pass the ball as soon as possible. Therefore, the pass frequency gets increase. As for the emerged behavior, we can find in Table 1 that the same tendency; (i.e., keepers do not pass the ball until takers become near to  $K_1$ ) under the takers' policy-(ii) as the case of taker's policy-(i).

## 5 Discussion

We show the difficulties in designing function for *keepaway* task through some experiments. Since *keepaway* is continuing and multiagent’s task, it is hard to decide when and what amount of reward will be given to the learners. Table 3 shows the comparison among three cases of *keepaway* ; i.e., hand-coded, and two reward functions. Though the empirical results show that our keepers can possess the ball for about three times longer than the cases of hand-coded and learned by another reward function[3], the reason of this high performance is not qualitatively analyzed yet.

Firstly, we discuss on reward functions that we introduced.

We introduce  $T = \text{EpisodeEndTime} - \text{LastActionTime}$  and give  $-1/T$  to the agent when task fails. Otherwise, agent receives 0 constantly during successful situation as mentioned in Section 3.4. The reason why sharing the same reward ( $= 0$ ) among the agents in the successful situation is that we could not tell which agent contributes the task from the length of keeping time. Since it is not always good for team that one agent keeps task longer locally, we did not introduce the predefined value of each agent’s reward individually. In our design, each agent will receive the reward  $f(T)$  according to its amount of time period: i.e., from taking off the ball (*LastActionTime*) to the end of task(*EpisodeEndTime*), at the end of each episode. Within the introduced reward functions,  $r_f = -1/T$  provides relatively better performance than that of other functions. Though function  $f(T) = -0.7^T$  draws a similar curve to  $r_f$  when  $T < 20$ , as shown in Figure 7, it doesn’t perform well as the case of  $r_f$ ’s. The main reason for this result is due to the value range of  $T$ . The range of  $T$  is always larger than 20, then the similarity in  $T < 20$  does not affect much on the performance.

Second, as for the frequency of pass, keeper with  $r_f$  passes the ball more frequently than the keepers with  $r_s$  in the earlier stage of learning, as shown Figure 12. Because keeper with  $r_s$  can get some reward when selecting HoldBall in early stage, the keeper becomes not to pass the ball so many times to other keeper. Meanwhile, our keepers, reinforced by  $r_f$ , do not get any penalty when they pass the ball, that is, they get penalty only when they are intercepted or missed the pass. So, our keepers do not afraid of pass to others. In the middle and late learning stages, the keepers with  $r_s$  become to pass the ball frequently because they experience more large reward using PassBall( $k$ ) than using HoldBall. Whereas the pass frequency of our keepers become decrease because they experience being intercepted the ball or missing the pass, after considerable training.

**Table 3.** Comparison of average possession times( in simulator seconds) for hand-coded and learned policies against always GoToBall takers in region 20[m] × 20[m]

Keeper Policy	[seconds]
Hand-coded by [3]	9.6
Learned by [3]	10.4
Learned by Ours.	35.5 ± 3.8

Third, in regard to the visual information that we described in Section 2.2, it contains some noise. The passed ball often fails to reach intended destination of kicker's because of the noise, and the noise causes a large effect on the emerging behavior. Since action of 'pass' has some probability of missing or being intercepted, the frequency of pass of our keepers learned with  $r_f$  becomes small. It is considered reasonable and proper behavior in noisy environment and against takers' policy-(i) shown in Figure 2(i).

Forth, we look at the effects of taker's policy. We found in Figure 13 that our keepers with  $r_f$  against takers' policy-(ii) (Figure 2(ii)) keep possess the ball less than in the case against takers' policy-(i) (Figure 2(i)). It seems that as the frequency of pass increases, the duration of episode decreases. We found in Table 2 that the frequency of pass becomes smaller when our keepers learn about takers' policy-(ii) than the case of taker's policy-(i).

Finally, we would like to discuss macro-actions we currently used. As increasing the pass frequency, keepers can not have enough time to move to the position that is free from the opponent, and can not open the way to let a ball pass from the current position of the ball. Consequently probability of missing a pass seems to be increasing. It might be resolved by introducing more sophisticated macro-action such as **GetOpen()** and so forth.

## 6 Conclusion

In this paper, we discuss reward design issue for multiagent continuing tasks and introduce the effective reward function for *keepaway* domain.

Though our experimental results show better performance than the previous works in *keepaway*, we do not give any theoretical analysis on the results yet. At present, we examine the problem peculiar to *a continuing task* that terminates at failure situation, such as a pole balancing, and a reward assignment within *a multiagent task* where simultaneous learning. For the continuing case, we show that a certain penalty would make agent learn successfully. While, regard with the multiagent case, we avoid the harmful effect of agents' simultaneous learning by means of parameter tuning. We would like to show a guideline for parameter tuning, and find a clue to resolve a reward assignment within multiagent reinforcement learning context in the future.

## References

1. Peter Stone, R.S.S.: Keepaway soccer: a machine learning testbed. In Andreas Birk, Silvail Coradeschi, and Satoshi Tadokoro, editors RoboCup-2001: Robot Soccer World Cup V. (2002) pp.214–223
2. Gregory Kuhlmann, P.S.: Progress in learning 3 vs. 2 keepaway. In Proceedings of the RoboCup-2003 Symposium (2003)
3. Peter Stone, Richard S. Sutton, G.K.: Reinforcement learning for robocup soccer keepaway. *Adaptive Behavior* **13**(3) (2005) 165–188

4. Stone, P., Kuhlmann, G., Taylor, M.E., Liu, Y.: Keepaway soccer: From machine learning testbed to benchmark. In Noda, I., Jacoff, A., Bredendfeld, A., Takahashi, Y., eds.: RoboCup-2005: Robot Soccer World Cup IX. Springer Verlag, Berlin (2006) To appear.
5. Ng, A.Y., Russell, S.: Algorithms for inverse reinforcement learning. In: Proc. 17th International Conf. on Machine Learning, Morgan Kaufmann, San Francisco, CA (2000) 663–670
6. Satinder P. Singh, R.S.S.: Reinforcement learning with replacing eligibility traces. *Machine Learning* **22**(1-3) (1996) 123–158
7. Richard S. Sutton, A.G.B.: Reinforcement Learning: An Introduction. A Bradford Book, The MIT Press (1998)



# Coordination of Concurrent Scenarios in Multi-agent Interaction

Rie Tanaka, Hideyuki Nakanishi, and Toru Ishida

Department of Social Informatics, Kyoto University  
Yoshida Honmachi, Sakyo-ku, Kyoto, 6068501, Japan  
+81-75-753-4820  
rtanaka@ai.soc.i.kyoto-u.ac.jp

**Abstract.** Though research on agents that interact with humans via voice or text has been intensively conducted, agents that can interact with more than two persons in parallel have not been studied well. To enable an agent to interact with multiple people, we propose a method to assign multiple scenarios to one agent, in which each scenario describes an interaction protocol between the agent and one person. Obviously, coordination among multiple scenarios is required to avoid conflicts in actions. For example, one agent cannot execute both walking and sitting actions simultaneously. However, what is more important is that a coordination policy, a way of specifying how to manage conflicts among multiple actions, must be introduced. In this paper, we introduce a coordination scenario that avoids conflicts in actions and coordinates scenarios according to a coordination policy. The coordination scenario, which controls the execution of interaction scenarios by communication, is generated by a coordination policy for solving conflicts. When the coordination scenario receives a request to execute actions from an interaction scenario, it checks whether the actions will trigger conflicts and sends an order not to execute them if conflicts will occur. The coordination scenario interworks concurrent interaction scenarios by executing this process repeatedly.

**Keywords:** Agent and digital cities; Conflict resolution and negotiation; Agent communication languages, dialog & interaction protocols; Multi-agent systems and their applications.

## 1 Introduction

Many studies on agents interacting with humans by voice or text have been conducted. In studies on embodied conversational agents[Cassell 00], agents embodied in a virtual space that use verbal and nonverbal communications have been constructed. For example, Cosmo[Lester 00], a life-like pedagogical agent, teaches about networks one-on-one by pointing to objects, while Steve[Rickel 99b], a task-oriented pedagogical agent, interacts with avatars controlled by users and achieves general tasks in the virtual world.

This research aims to enable an embodied agent to successfully interact with multiple people by verbal and nonverbal communication. We note that not all interactions in the real world are just one-on-one, and the purposes of interaction or the roles in interaction are not always obvious. Humans have different ways of interacting with different persons; we call them by the general term interaction protocols. When interacting with multiple people, protocols are chosen to suite each person. Therefore, in this paper, rather than plural plans or rules, we give plural protocols to each agent to achieve the entire goal. We control agents by scenarios, which describe the interaction protocols in an executable form.

To merge multiple protocols into single scenario is difficult. This is because all combinations of all possible behaviors of each person should be described for the purpose of enabling the agent to interact with multiple persons successfully even if they behave individually, i.e. not cooperatively. Therefore, we describe one protocol in each scenario and execute the scenarios concurrently. Since problems occur when we execute multiple scenarios concurrently, we propose a method that coordinates concurrent scenarios so as to execute them successfully.

## 2 Problems with Concurrent Execution

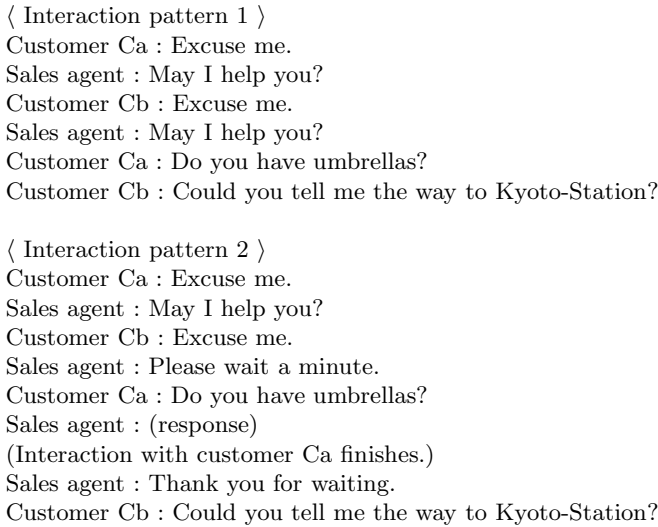
Assume that a sales agent has scenario Sa for interaction with customer Ca and scenario Sb for interaction with customer Cb. Interaction which occurs between the agent and customer Ca and between the agent and customer Cb is shown in Figure 1. When customers Ca and Cb contact the agent, there are several interaction patterns (two of them are shown in Figure 2). As shown by these examples, there are various ways of interacting with multiple persons.

To realize the interaction shown in Figure 2, it is necessary to solve two problems. One problem is the conflicts in actions. For example, the sales agent tries to speak to customers Ca and Cb simultaneously if they approach the agent at the same time. Since the sales agent has only one body, such an action is physically impossible. The other problem is that we need to design a coordination policy to coordinate scenarios. A coordination policy specifies how to manage

⟨ Interaction using scenario Sa ⟩  
 Customer Ca : Excuse me.  
 Sales agent : May I help you?  
 Customer Ca : Do you have umbrellas?

⟨ Interaction using scenario Sb ⟩  
 Customer Cb : Excuse meD  
 Sales agent : May I help you?  
 Customer Cb : Could you tell me the way to Kyoto-Station?

**Fig. 1.** Examples of interaction with a single person



< Interaction pattern 1 >  
Customer Ca : Excuse me.  
Sales agent : May I help you?  
Customer Cb : Excuse me.  
Sales agent : May I help you?  
Customer Ca : Do you have umbrellas?  
Customer Cb : Could you tell me the way to Kyoto-Station?

< Interaction pattern 2 >  
Customer Ca : Excuse me.  
Sales agent : May I help you?  
Customer Cb : Excuse me.  
Sales agent : Please wait a minute.  
Customer Ca : Do you have umbrellas?  
Sales agent : (response)  
(Interaction with customer Ca finishes.)  
Sales agent : Thank you for waiting.  
Customer Cb : Could you tell me the way to Kyoto-Station?

**Fig. 2.** Examples of interaction with multiple persons

conflicts, in other words, how to coordinate scenarios. In the above example, the coordination policy would specify which person the agent responds first. It is necessary to define the method to describe the coordination policy and reflect it to the coordination scenario.

## 3 Scenario Coordination Architecture

### 3.1 Coordination Scenario

This research assumes that each agent has several scenarios. In related research, in the area of multi-agent planning, Georgeff suggested the method of coordination among agents, each of whom has a plan [Georgeff 83]. He suggested that communication functions are added to each plan, and plans are coordinated by the synchronization program. In this research, we propose the use of a coordination scenario to coordinate interaction scenarios.

The coordination scenario controls the execution of actions by communicating with each interaction scenario. A protocol consists of pairs of an event and corresponding sequential multiple actions, so the coordination scenario controls action sequences. The coordination scenario obeys the coordination policy described by the scenario writer. Furthermore, it has functions of detecting and solving conflicts in the action sequences. The coordination scenario examines whether conflicts can occur, and if conflict is possible, it orders interaction scenarios not to execute the offending action sequences.

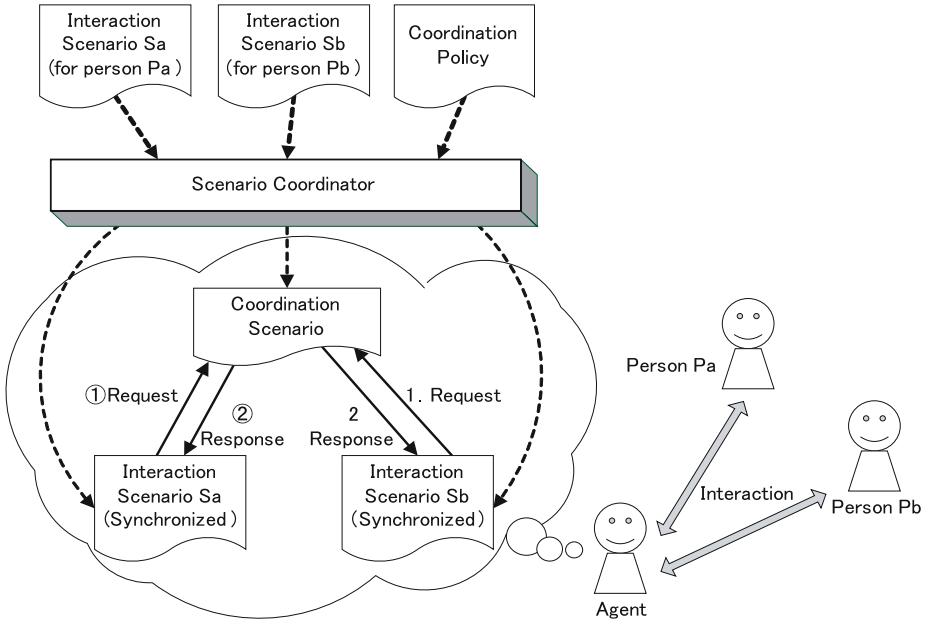


Fig. 3. Scenario coordination architecture

Figure 3 shows the whole process of coordination. The scenario coordinator automatically generates the coordination scenario from the coordination policy and interaction scenarios. It also changes the interaction scenarios so as to communicate with the coordination scenario. Every time an event is observed and actions corresponding to it are to be executed in the changed interaction scenario, the coordination scenario controls the scenario in the following manner. 1) Changed interaction scenarios send requests to the coordination scenario before executing an action sequence and wait for a response. 2) The coordination scenario examines whether a requested action sequence conflicts with the action sequences being executed, and gives an order to execute it only if it doesn't conflict with any of action sequences. 3) Each interaction scenario obeys the response of the coordination scenario.

### 3.2 Scenario Coordinator

The scenario coordinator changes the interaction scenarios if necessary and generates the coordination scenario. As for the interaction scenarios, the scenario coordinator adds communication functions to each scenario which describes one-on-one interaction.

The coordination scenario is generated by the scenario coordinator from both the coordination policy and interaction scenarios. At first, the scenario coordinator examines all combinations of the action sequences used in all interaction

scenarios and if pairs of action sequences are found to conflict, the pairing is stored as conflict data. When the scenario coordinator detects conflicts in action sequences, it refers to the definition of conflict conditions, which we define as follows. The scenario coordinator refers to the coordination policy when generating the coordination scenario so that the functions in the coordination scenario refer to the conflict data.

We define conflict conditions using precondition, postcondition, continuance condition, and the resource of the action sequence. The continuance condition should be satisfied during execution of an action sequence that takes some time, for example, talking action or walking action. The resource means, for example, the legs or hands of the agent. There are five conflict conditions as follows. The scenario coordinator judges that conflict will occur if any one of the conditions is satisfied.

- 1) The concurrent execution of the action sequences yields a result that differs from the result of any sequential execution.
- 2) Continuance conditions of the action sequences contradict each other.
- 3) The continuance condition of an action sequence currently being performed makes it impossible to satisfy the precondition of the next action sequence.
- 4) The continuance condition of one action sequence contradicts the postcondition of the other.
- 5) The action sequences use the same resources.

## 4 Scenario Coordination Mechanism

### 4.1 Generation of the Coordination Scenario

We describe the coordination policy in the same format as the interaction scenarios. The reason is as follows. The coordination scenario coordinates interaction scenarios by controlling execution of the actions through communication. In other words, the coordination scenario repeats the process of receiving

**Table 1.** The meanings of messages

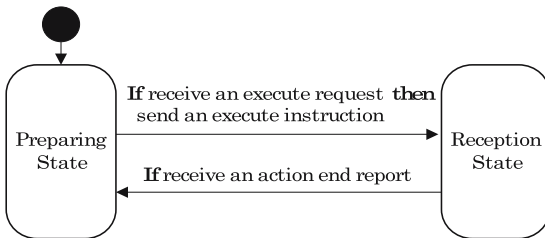
Request Messages	Meanings
execute request	Order to execute an action sequence
action end report	Report the end of an action sequence
wait report	Report the end of the before-wait procedure
scenario end report	Report the end of execution of a scenario

Response Messages	Meanings
execute instruction	Order to execute an action sequence
annul instruction	Order not to execute an action sequence
wait instruction	Order not to execute an action sequence and execute the before-wait procedure

a request message, such as a request to execute an action sequence, from an interaction scenario and sending a response message to the request. Since interaction scenarios describe observations of the environment and actions corresponding to the observation results, this format is suitable for the coordination policy.

In the coordination policy, request messages sent by interaction scenarios are the observed target, and the sending of response messages from the coordination scenario is equivalent to the actions corresponding to the observation results. Table 1 shows typical request and response messages. In the messages, the wait report and the wait instruction are used in the wait procedure, which is initiated when the agent wants to keep a person waiting. The wait procedure is executed as follows. 1)The coordination scenario sends a wait instruction to the interaction scenario associated with the person who is to be kept waiting. 2)The interaction scenario executes the before-wait procedure, for example, saying "Please wait a minute" to inform the person of the agent's intention. 3)The interaction scenario sends a wait report to the coordination scenario and waits for the order to resume. When the interaction scenario receives an execute instruction, which means the order to resume, it executes the after-wait procedure, for example, saying "Thank you for waiting". After execution of the after-wait procedure, it proceeds with the interaction. In this process, the behaviors used in the before-wait procedure and the after-wait procedure can be freely set.

< Coordination policy : Corresponding to pattern 1 >



< Coordination policy : Corresponding to pattern 2 >

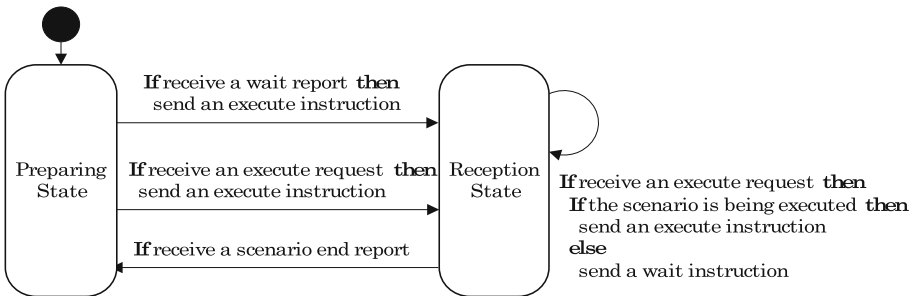


Fig. 4. Examples of the coordination policy

```

x ← the action sequence
S ← list of action sequences being executed

If the execute instruction corresponds to the execute request then
  If S is empty then send an execute instruction
else
  for each s in S do
    if x conflicts with s then
      send an annul instruction
    return
  end
  send an execute instruction
else if the execute instruction corresponds to the wait report then
  If S is empty then send an execute instruction
else
  result ← false
  loop do
    for each s in S do
      if x conflicts with s then result ← true
    end
    if result = false then
      send an execute instruction
    return
  end

```

**Fig. 5.** Algorithm to send an execute instruction

Figure 4 shows the coordination policies following the two patterns in Figure 2. It is described by using the state machine model. The policy corresponding to pattern 1 means that the coordination scenario, which is generated from the policy, sends an execute instruction every time an action sequence is requested without consideration of which scenario sent the request.

The policy corresponding to pattern 2 means that the agent keeps a person waiting until the current interaction finishes. Specifically, since one scenario describes interaction with one person, the coordination scenario sends an order to the second scenario to wait until execution of the first scenario finishes. At first, when a person coarrives and the corresponding scenario sends an execute request, the coordination scenario sends an execute instruction and transits to the reception state. When the coordination scenario receives an execute request, it sends an execute instruction to the scenario being executed, or a wait instruction to any other scenario. After execution of the scenario finishes, the coordination scenario transits to the preparing state. If it receives a wait report, which means there are one or more scenarios waiting, it sends an execute instruction to resume the execution and transits to the reception state. The coordination scenario continually repeats this process.

The interaction scenario consists of several rules. We represent the rule as below.

**If** *event* is observed **then** execute *actions*

We call the part “**If** *event* is observed **then**” as the conditional part, and the part “execute *actions*” as the action part.

The scenario coordinator loads a scenario from a file.

S ← scenario

D ← empty list

**for each** *rule* **in** S **do**

    L, C ← empty list

    Add character strings which mean following commands or part to the end of C

        A command “send an execute request to the coordination scenario”

        A command “wait for response”

        A conditional part “**If** the response = an execute instruction **then**”

        The action part of *rule*

        A command “send an action end report to the coordination scenario”

        A conditional part “**else if** the response = a wait instruction **then**”

        A command “execute the before-wait procedure”

        A command “send a wait report to the coordination scenario”

        A command “wait for an execute instruction”

        A command “execute the after-wait procedure”

        The action part of *rule*

        A command “send an action end report to the coordination scenario”

        A conditional part “**else if** the response = an annul instruction **then**”

        A command “memorize that *event* was observed”

        A command “send an action end report to the coordination scenario”

        A command “continue observation”

    Add the conditional part of *rule* to the end of L

    Add C to the end of L

    Add a character string which means a conditional part

    “**else if** it is memorized that *event* was observed **then**” to the end of L

    Add the copy of C to the end of L

    Add L to the end of D

**end**

Write each element of the list D into a new file in order

**Fig. 6.** Algorithm to add functions to interaction scenarios

The coordination scenario is generated simply by changing the observation parts and action parts in the coordination policy into executable functions. However, since no description of the conflicts is included in the coordination policy, the scenario coordinator generates the coordination scenario by adding functions for examining the conflicts to the changed coordination policy. Functions are added to the part that sends the execute instructions. Figure 5 shows the algorithm used to send an execute instruction. There are two kinds of situations in which an execute instruction is sent. One is the case when the coordination



```

If event is observed then
  Send an execute request to the coordination scenario
  Wait for response
  If the response = an execute instruction then
    Execute actions
    Send an action end report to the coordination scenario
  else if the response = a wait instruction then
    Execute the before-wait procedure
    Send a wait report to the coordination scenario
    Wait for an execute instruction
    Execute the after-wait procedure
    Execute actions
    Send an action end report to the coordination scenario
  else if the response = an annul instruction then
    Memorize that event was observed
    Send an action end report to the coordination scenario
    Continue observation
else if it is memorized that event was observed then
  Execute the same procedure executed when event was observed

```

**Fig. 7.** Synchronization of interaction scenarios

scenario sends order corresponding to the execute request. The other is the case when it sends order to the scenario which keeps waiting for the execute instruction to resume interaction. In the latter case, the coordination scenario keeps waiting during the action sequence to be resumed conflicts with the action sequences being executed. This is because, the scenario which has sent a wait report keeps waiting for an execute instruction and it is impossible to annul the resumption.

## 4.2 Synchronization of Interaction Scenarios

The scenario coordinator adds functions of sending request messages to the coordination scenario and following response messages to each interaction scenario. The interaction scenario consists of several rules. Figure 6 shows the algorithm used to add functions to each rule. Figure 7 shows the rule after adding functions in accordance with the algorithm. If an interaction scenario receives an execute instruction, it simply obeys the instruction. However, if it receives an annul instruction, it not only obeys it, but also memorizes that the event was observed. After that, if it observes a new event, it sends a request to execute the new action sequence. If no new event is observed, it sends a request to execute the action sequence corresponding to the memorized event again.

By following the above procedure, an interaction scenario can repeatedly send the same request until it receives the execute instruction, except the case when

the situation has changed. If the situation has changed, namely, if it is needed to execute new action sequence corresponding to new event instead of memorized event, the interaction scenario can annul the memorized event and send a new request. This indicates that the interactions scenario can meet the change of situations.

## 5 Implementation

We realized our proposal by using FreeWalk/ $Q$ [Nakanishi 04]; it allows us to control agents by scenarios which describe interaction protocols. In FreeWalk/ $Q$ , protocols are described in  $Q$  language, and are called  $Q$  scenarios. To describe a scenario, we use defined cues which mean an event that triggers interaction and actions which are behaviors corresponding to cues. Therefore, we used the state machine model to describe the coordination policy as well as scenarios, and defined cues and actions that are used to describe the coordination policy. We implemented the scenario coordinator which generates the coordination scenario and changes the interaction scenarios as described above.

Figure 8 displays a screen shot of FreeWalk/ $Q$ . The woman at the center is a sales agent and has the coordination scenario and changed scenarios. We use the coordination policy corresponding to pattern 2, which is shown in Figure 4. During the execution of scenarios, she first interacts with the male customer on the right. When the female customer attempts to talk her, she says "Please wait

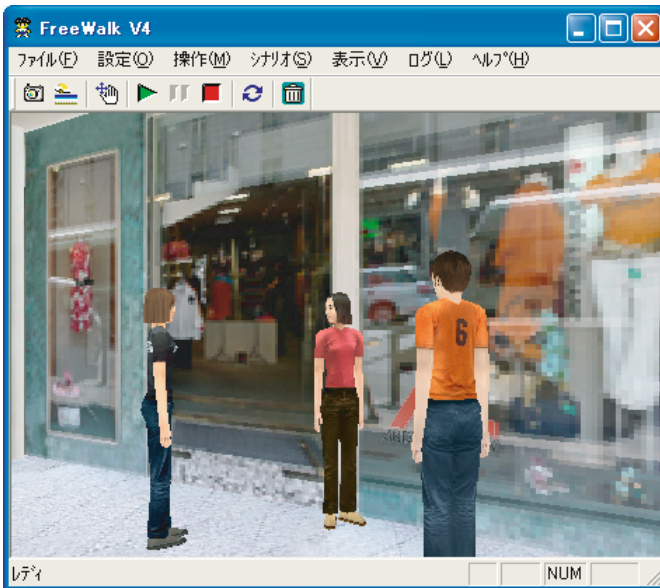


Fig. 8. Screen shot of FreeWalk/ $Q$

a minute” and keeps her waiting. When the interaction with the male customer finishes, she turns to the female customer and says ”Thank you for waiting.”

## 6 Conclusion

This research aims to enable agents to successfully interact with multiple people. Specifically, we assume that multiple interaction protocols are assigned to one agent. We design one interaction scenario for each interaction protocol and assign several interaction scenarios to one agent. The agent executes scenarios concurrently. If we execute multiple scenarios concurrently, we must prevent conflict between the multiple actions, and a coordination policy that describes how to manage conflicts is needed. For coordination, we introduce the coordination scenario; it communicates with the interaction scenarios to control the execution of actions.

The coordination scenario is generated by adding functions of detecting and solving conflicts to the coordination policy. We propose a scenario coordination architecture, in which the coordination scenario is generated and interaction scenarios are changed so as to communicate with the coordination scenario automatically. Generated scenarios are assigned to the agent, and the coordination scenario coordinates the interaction scenarios at run time. Furthermore, we implemented the architecture and verified that the agent behaves according to the coordination policy set.

In this paper, we propose a scenario coordination mechanism for concurrent execution. At run time, the interaction scenario sends a request to execute actions to the coordination scenario, and the coordination scenario sends orders to the interaction scenarios to execute them or not. The coordination scenario determines if the actions can be executed or not by referring to the five types of conflicts which we have newly defined.

The method proposed in this paper allows us to describe a policy without knowing the contents of interaction scenarios in detail. The innovation of this research is that interaction scenarios described by different persons can now be coordinated quite easily.

## Acknowledgement

This work was supported by a JSPS Grant-in-Aid for Scientific Research (17650041) and the International Communication Foundation.

## References

- [Cassell 00] Cassell, J., Sullivan, J., Prevost, S. and Churchill, E.: *Embodied Conversational Agents*, MIT Press (2000).
- [Lester 00] Lester, J. C., Towns, S. G., Callaway, C. B., Voerman, J. L. and FitzGerald, P. J.: Deictic and Emotive Communication in Animated Pedagogical Agents, In Cassell, J., Sullivan, J., Prevost, S. and Churchill, E.: *Embodied Conversational Agents*, MIT Press, pp. 123–154 (2000).

- [Rickel 99b] Rickel, J. and Johnson, W. L.: Virtual Humans for Team Training in Virtual Reality, In *Proceedings of the Ninth International Conference on AI in Education*, pp. 578–585, IOS Press (1999b).
- [Nakanishi 04] Nakanishi, H. and Ishida, T.: FreeWalk/*Q*: Social Interaction Platform in Virtual Space, *ACM Symposium on Virtual Reality Software and Technology (VRST2004)*, pp. 97–104 (2004).
- [Ishida 02] Ishida, T.: *Q*: A Scenario Description Language for Interactive Agents, *IEEE Computer*, Vol. 35, No. 11, pp. 54–59 (2002).
- [Georgeff 83] Georgeff, M.: Communication and interaction in multiagent planning, In *Proceedings of the 3th National Conference on Artificial Intelligence*, pp. 125-129 (1983).

# A Multi-agent Negotiation Model Applied in Multi-objective Optimization

Chuan Shi<sup>1,2</sup>, Jiewen Luo<sup>1,2</sup>, and Fen Lin<sup>1,2</sup>

<sup>1</sup> Key Laboratory of Intelligent Information Process  
Institute of Computing Technology Chinese Academy of Science

<sup>2</sup> Graduate University of the Chinese Academy of Sciences  
shic@ics.ict.ac.cn

**Abstract.** Although both multi-objective optimization and agent technology gained a lot of interest during the last decade, many aspects of their functionality still remain open. This paper proposes the multi-agent negotiation model applied in multi-objective optimization. There are three types of agents in the system. The plan agent plans the global best benefit; the action agent plans the best benefit of the single objective; and the resource agent manages the common resource. The agents compete and cooperate to reach the global best benefit through their negotiation. The model is applied in evolutionary multi-objective optimization to realize its parallel and distributed computation, and the experiment on MAGE shows the model is effective.

## 1 Introduction

With recent growth of need for faster and more reliable systems, interest has grown towards concurrent distributed systems. Multi-agent systems (MAS), in which independent software agents interact with each other to achieve common goals, complete concurrent distributed tasks under autonomous [1, 2]. In MAS, the negotiation not only strengthens agents and MAS's ability to solve problems, but also make the system more flexible to apply in more real problems.

Multi-objective problem is the popular problems in the scientific research and real projects. Evolutionary computation applying in multi-objective problem can provide a set of solutions in one run; the method is called evolutionary multi-objective optimization (EMO). Many multi-objective evolutionary algorithms (MOEA) have been proposed [3].

In this paper, we attempt to apply MAS in EMO. In real world, many agents are competitive to reach their best benefit; at the same time, they are cooperative to reach the global best benefit, since the resource is common and limited. There are three types of agents in the system. We define them in a uniform way, and then propose their negotiation model. Moreover, we apply this negotiation model in the evolutionary multi-objective optimization to demonstrate how to apply this model. At last we do a simulating experiment on MAGE [4], which proof the negotiation model is effective.

The rest of the paper is organized as follows: in the following section, we review the related work. Section 3 illustrates the multi-agent negotiation model in detail; and then the model is applied in the evolutionary multi-objective optimization in the section 4; at last, the section 5 makes a conclusion.

## 2 Related Work

This section introduces some related works. We first introduce multi-agent system, and then present some notation used in multi-objective optimization. At last we discuss some work on the combination of multi-agent system and multi-objective optimization.

### 2.1 Multi-agent System

The growth in networked information resources requires information systems that can be distributed on a network and interoperate with other systems. Such systems cannot be easily realized with traditional software technologies because of the limits of these technologies in coping with distribution and inter-operability. The agent-based technology seems be a promising answer to facilitate the realization of such systems because they were invented to cope with distribution and interoperability [5]. Recently, natural evolution of agent-based technology has led them to migrate from the research laboratories to the software industry. This migration is good news for the entire agent-based technology community, but it also leads to new expectations and new questions about agent-based technology such as MAS methodologies, tools, platforms, reuse, specification and etc. And MAS gains more and more interest in both the research area and the industry [6]. The best example is the brunch of announcements about new multi-agent platforms, such as OAA [7], RETSINA [8] and etc.

MAGE is a Multi-AGENT Environment with a collection of tools supporting the entire process of the software engineering based on agents, which is proposed by Prof Shi [3]. It is designed to facilitate the rapid design and development of new multi-agent applications by abstracting into a toolkit the common principles and components underlying many multi-agent systems. The idea was to create a relatively general purpose and customizable toolkit that could be used by software users with only basic competence in agent technology to analyze, design, implement and deploy multi-agent systems.

From fig.1, we can see that MAGE platform consists of the following parts, agent supporting environment and agent development environment. Agent development environment consists of modeling tool AUMP and design and programming tool VASstudio. AUMP is designed for system analysis and design stages. VASstudio is for system design, development and deployment stages. The workflow of MAGE platform is as the arrow shows. We use AUMP to get the system model and then design and programming it in VASstudio, finally we run the multi-agent system under the MAGE agent supporting environment. MAGE has been used by a number of companies and institutions. Further details and documentation can be found at <http://www.intsci.ac.cn/agent/mage.html>.

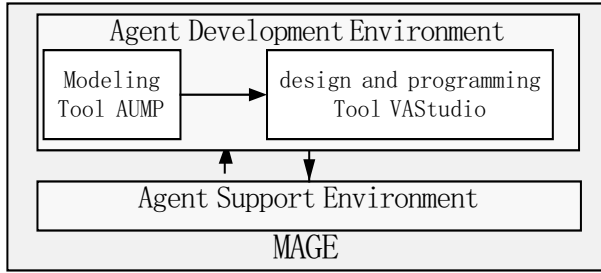


Fig. 1. MAGE Framework

### 2.2 Evolutionary Multi-objective Optimization

Multi-objective optimization problems (MOPs) are those problems that involve simultaneous optimization of more than two objectives (often competing) and usually there is no single optimal solution. It is usually difficult or even impossible to assign priorities as in single objective optimization problems. This makes an algorithm returning a set of promising solutions preferable to an algorithm returning only one solution based on some weighting of the objectives. For this reason, there has been an increasing interest in applying evolutionary algorithm (EA) to MOPs in the past ten years.

Veldhuizen and Lamont have rigorously defined multi-objective optimization problems and certain related concepts in [3]. There are two basic notions. Without loss of generality, we only consider the minimization optimization problems in this paper, since it is easy to change maximization problems to minimization problems.

**Definition 1** [3]. General MOP: an MOP minimizes  $F(\mathbf{x}) = (f_1(\mathbf{x}), \dots, f_m(\mathbf{x}))$  subject to  $g_i(\mathbf{x}) \leq 0, i = 1, \dots, k$ ,  $\mathbf{x} \in \Omega$  ( $\Omega$  is the decision variable space). An MOP solution minimizes the components of the m-dimension objective vector  $F(\mathbf{x})$ , where  $\mathbf{x} = (x_1, \dots, x_n)$  is an n-dimension decision variable vector from some universe  $\Omega$ .

**Definition 2** [3]. Pareto dominance: If a vector  $U = (u_1, \dots, u_m)$  Pareto dominates  $V = (v_1, \dots, v_m)$ , denotes as  $U \preceq V$ , that is  $U \preceq V$  if and only if

$$\forall i \in \{1, \dots, m\} \quad u_i \leq v_i \wedge \exists i \in \{1, \dots, m\} \quad u_i < v_i. \tag{1}$$

Most contemporary research on MOP is based on Pareto dominance. A decision vector  $x_u \in \Omega$  is said Pareto optimal if and only if there is no  $x_v \in \Omega$  for which  $F(x_v) \preceq F(x_u)$ . The set of all Pareto optimal decision vectors is called the Pareto optimal set of the problem. The corresponding set of the objective vector is called the nondominated set, or Pareto front.

Over the past decade, a number of multi-objective evolutionary algorithms have been suggested [3, 9]. These MOEAs use Pareto dominance to guide the search, and return a set of nondominated solutions as result. Unlike in single-objective optimization, there are two goals in a multi-objective optimization: 1) convergence to the Pareto optimal set and 2) maintenance of diversity in solutions of the Pareto optimal set [9].

### 2.3 Multi-agent System Applied in Evolutionary Multi-objective Optimization

Multi-agent system attempts to simulate the rational behavior of human being by agent, to describe the system not through certain algorithm, but through rational interaction among agents. MAS is composed of a number of agents capable of communicating, coordinating, cooperating and even competing with each other. Evolutionary multi-objective optimization utilizes the population evolutionary to reach the optimal value. Evolutionary algorithm attempts to simulate the nature's evolutionary process, which obtains the more optimal solutions through reproduction, crossover, and mutation. These two approaches both have the colony characteristic and naturally parallel characteristic, so the combination of these two approaches is a nature idea to solve some problems.

Some researchers have done work in this aspect. An evolutionary multi-agent system (EMAS) is proposed to solve multi-objective optimization [10]. The key idea of EMAS is the incorporation of evolutionary processes into MAS at a population level. It means that besides interaction mechanisms typical for agent-based systems, agents are able to reproduce and may die. A decisive factor of the agent's activity is its fitness, expressed by the amount of possessed non-renewable resource called life energy. Selection is realized in such a way that agents with high energy are more likely to reproduce, whereas a low level of energy increases possibility of death. In fact all decisions about actions to be performed are made autonomously by agents, and thus EMAS may be considered as a computational technique utilizing a decentralized model of evolution, unlike classical evolutionary computation.

Naitoh and Terano describe an agent-based simulation model for analyzing corporate behaviors of competing firms [11]. The simulator employs evolutionary computation to evolve the characteristics of the firms. The method also addresses a novel technology of agent-based modeling with genetic algorithms.

Vacher and Galinho use a multi-agent system guided by a multi-objective genetic algorithm to find a balance point in the respect of solution of the Pareto front [12]. By crossover and mutation of agents, according to their fitness function, the method can improve the existing solution.

Cetnarowicz propose autonomous agents as an extension to the object and process concepts [13]. The active agent is invented as a basic element of which distributed and decentralized systems can be built. The use of evolution strategies in design of multi-agent systems reveals new possibilities of developing complex software systems. The evolution plays a key role in creation and organization of social structures.

## 3 A New Multi-agent Negotiation Model

Multi-objective optimization is the popular problems existing in real world. The benefit of a colony can be expressed as an objective function. The different colony's benefits constitute the multi-objective functions. Because the different colonies use the common and limited resources, the multi-objective functions usually are conflict, however, the whole system must obtain the global best benefit. How to obtain the global optimal solutions? Each colony attempts to reach its best benefit. However,



they all use the common resource and the resource is limited, they must negotiate with each other and reallocate the resource to obtain the best global benefit.

Through the analyses above, we find there are three types of agents in the system from the multi-agent system point of view: resource agent (RA), action agent (AA), and plan agent (PA). The resource agent stands for the common resource, which allocates the resource to the different the action agents. The action agent stands for the single objective, which attempts to obtain the best benefit of a colony. The plan agent stands for the agent that harmonizes the benefit of the different colonies to reach the global best benefit. They can be defined in a uniform way.

**Definition 3.** A negotiating agent model is a five tuple  $\langle \mathcal{K}, \mathcal{A}, \mathcal{G}, \mathcal{P}, I \rangle$ .

$\mathcal{K}$  is the agent's belief knowledge base.

$\mathcal{A}$  is the agent's action ability set.

$\mathcal{G}$  is the agent's goal set.

$\mathcal{P}$  is the agent's plan base.

$I$  is the agent's communication.

There are three types of agents in the multi-agent system. They have different functions and goals. They can be defined as the following format.

**Definition 4.** A plan agent (PA) is the five tuple  $\langle \mathcal{K}, \mathcal{A}, \mathcal{G}, \mathcal{P}, I \rangle$  that plan the global objective and negotiate with different action agents.

$\mathcal{K}$  includes the using information of the resources and the development of each objective.

$\mathcal{A}$  includes the plan of the global objective (the function is `ObjectivePlan()`) and sending the result to RA (the function is `SendResPlan(RA, ResInf)`).

$\mathcal{G}$  is to reach the best benefit of the global objective.

$\mathcal{P}$  plans the global objective and finds the optimal strategy according to the information in  $\mathcal{K}$ .

$I$  communicate with AA and PA.

**Definition 5.** An action agent (AA) is the five tuple  $\langle \mathcal{K}, \mathcal{A}, \mathcal{G}, \mathcal{P}, I \rangle$  that plan the single objective.

$\mathcal{K}$  includes the resource information that the agent uses.

$\mathcal{A}$  include the action that plan the single objective (the function is `SingleObjPlan()`) and send the agent's benefit information to PA (the function is `SendObjBenefit(PA, Inf)`).

$\mathcal{G}$  includes optimizing the single objective.

$\mathcal{P}$  is  $\Phi$ , namely there are no global objective plan.

$I$  communicate with PA and RA.

**Definition 6.** A resource agent (RA) is the five tuple  $\langle \mathcal{K}, \mathcal{A}, \mathcal{G}, \mathcal{P}, I \rangle$  that manage and allocate the resource.

$\mathcal{K}$  includes the resource information and its using status.

$\mathcal{A}$  include the actions that allocate the resource (the function is `ResourcePlan()`) and send the allocated resource information to AA (the function is `SendResInf(AA, ResInf)`).

$\mathcal{G}$  includes finding the optimal resource management.

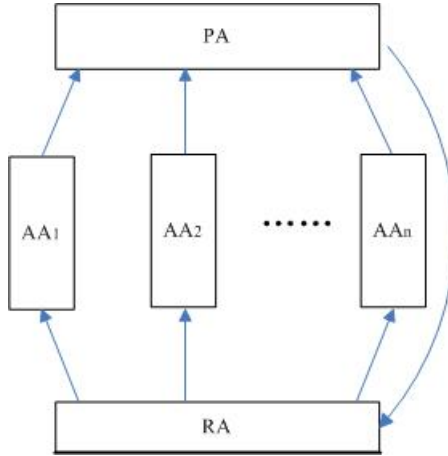


Fig. 2. The demonstration of MOMAS

```

RA.Initial(); //Initialize RA
PA.Initial(); //Initialize PA
AA.Initial(); //Initialize AA
Do
{
    RA.ResourcePlan(); //RA allocate the resource according to the planning
                        information.
    For each AAi
        RA.SendResInf(AAi, ResInf); //RA send the resource information to each AAi.
    Parallel:
        AAi.SingleObjPlan(); //each AAi plans its single objective.
        AAi.SendObjBenefit(PA, Inf); //each AAi send its best benefit to PA.
    PA. ObjectsPlan(); //PA plan the global best benefit.
    PA.SendResPlan(RA); // PA send the planning information to RA.
} While (stopping criterion is satisfied);
    
```

Fig. 3. The negotiation algorithm of MOMAS

$\mathcal{P}$  is  $\Phi$ , namely there are no global objective plan.  
*I* communicate with PA and AA.

The multi-agents system cooperates to accomplish a multi-objective optimal task. The agents negotiate in the following steps: RA allocates the resources (the resources are allocated randomly at first) and sends them to AA; AA optimizes its own objective with its allocating resource, and then AA sends its optimized result to PA. According to the AAs' result and the global goal, PA plan the global objective with some rules, and send the planning information to RA. According to the planning information, RA real-locates the resource and sends them to AA. The process is continued until the stopping

criterion is satisfied. The multi-agent system applied in multi-objective optimization is called MOMAS. The workflow of the MOMAS is demonstrated in Fig.2. We describe the MOMAS' negotiation algorithm in Fig.3.

In the description above, we use some abstract notions, for example global objective, information, plan etc. They have different meanings for different applications.

## 4 Application in Evolutionary Multi-objective Optimization

In this section, we will demonstrate how MOMAS is applied in the evolutionary multi-objective optimization.

The resource in RA is the individuals in the population. RA generates the new population according to the information from the PA and sends the individuals to each objective function. The objective in AA is the objective function. It evaluates the fitness of the individuals in the population. The plan task in PA is sorting these individuals according to their fitness, and sends this information to RA.

The negotiation among agents is as following steps. The RA manages the population, and sends the information of the individuals to the objective functions. The objective functions in AA evaluate the individuals at the same time, and then they send their results to the PA. The PA sorts these individuals according to their fitness, and sends the information to RA. According to the rank information of individuals, the RA eliminates some individuals and generates some new individuals to form the new-generation population, and send these individuals to the AA. The loop is continued until the stopping criterion is satisfied. The negotiation of agents can be seen in fig.4 From the analyses above, we find that we divide the MOEA into three main parts: the population maintenance, fitness evaluation, and nondominated sorting of the population. These three parts use the different agents. So the MOMAS realize the parallel and distributed computation of MOEA.

From the analyses in MOMAS, we find it is different from EMAS. EMAS simulate the social system at a population level, namely each agent stands for an individual. However, MOMAS is at the objective level, in other words, an agent stands for an objective. So MOMAS has higher granularity as compared to EMAS. Vacher's system is at population level too. MOMAS simulating a popular social system, so it is general model and can be used in many aspects.

We do the simulating experiments to check how the model works in the evolutionary multi-objective optimization. The experiment is done on MAGE. The test function is the following test problem.

$$\begin{aligned} f_1(\mathbf{x}) &= x^2 \\ f_2(\mathbf{x}) &= (x-1)^2 \\ x &\in [-10, 10] \end{aligned} \quad (2)$$

Fig.5 show the function's figure and its Pareto front district. There are four agents in the experiments. One RA manages the population; two AA calculate the fitness of these two functions respectively; and the PA sorts the individuals according to their

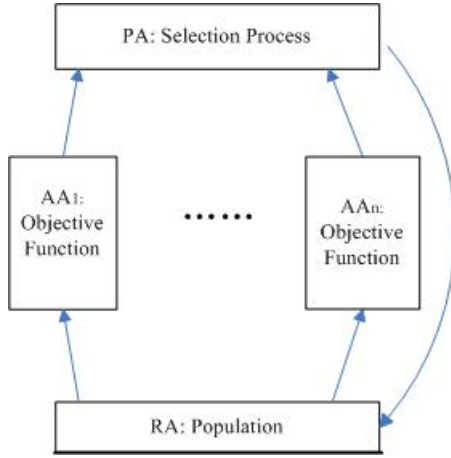


Fig. 4. The negotiation of MOMAS in evolutionary multi-objective optimization

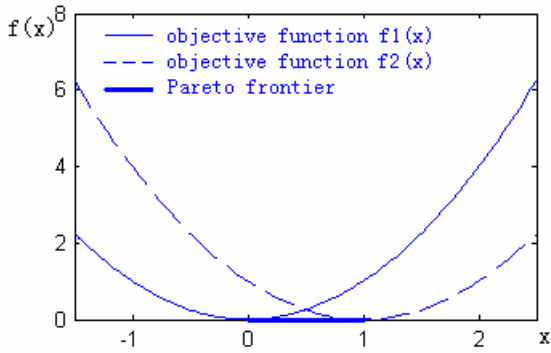


Fig. 5. Sample optimization problem and the Pareto frontier discovered by the system

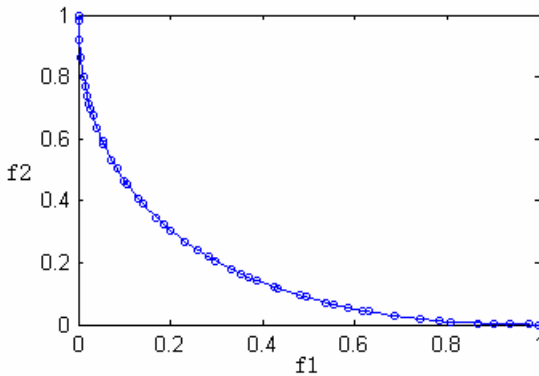


Fig. 6. The result of simulation experiment

fitness. The MOEA used in the experiment is NSGA-II [9]. The sorting algorithm used in PA is the nondominated sorting algorithm in NSGA-II, In RA, the child individuals is generated with simulated binary crossover (SBX) and polynomial mutation operator [9], the new-generation individuals is selected from the combination of the new-generated individuals and the individuals in the current generation according to their rank and density estimation. The population size is 50, and the running generation is 50. The result is shown in fig.6. We can observe that the solution converge to the Pareto optimal front and maintain the good diversity. The experiment proof the method is effective.

## 5 Conclusion

This paper proposes a new multi-agent negotiation model applied in multi-objective optimization--MOMAS. According to the analysis in the multi-objective optimal problem, agents are divided into three types. The plan agent plans the global best benefit, the action agent plans the best benefit of the single objective, and the resource agent manages the common resource. They compete with each other and cooperate to reach the global benefit at the same time. We use the negotiation model to realize the parallel and distributed computation of evolutionary multi-objective optimization. The population is used as the resource in the resource agent. The evaluation of the single objective is used as the action agent. The sorting process of individuals is seen as the global optimization in the plan agent. These three types of agents cooperate to reach the Pareto optimal solutions. An experiment on MAGE has proofed the model is effective.

In this paper, the emphasis is laid on the ideal and method. As a parallel and distributed computation method, MOMAS has many advantages to apply in the evolutionary multi-objective optimization. Some multi-objective optimal problems' function evaluation is time-consuming, for example job scheduling problem; some problems have many objectives. MOMAS can parallel evaluate these objectives in different computers at the same time, which will speed up MOEA significantly. We will solve more complex problems with MOMAS to explore its potential in the further work.

## Acknowledgments

This work is supported by the National Science Foundation of China No. 60435010, National Basic Research Priorities Programme No. 2003CB317004 and the Nature Science Foundation of Beijing No. 4052025.

## References

1. David, N.A., etc. Multiagent cooperation in International coalitions. *Intelligent System*, 26-35, May, 2002.
2. Shi, Z.Z. *Intelligent agent and its application*. Science Publishing Company, China.
3. Veldhuizen, D.A.V., Lamont, G.B.. *Multi-objective Evolutionary Algorithms: Analyzing the State-of-the-Art*. *Evol. Comput.*, vol.18, no.2, pp.125-147, 2000.

4. Shi, Z.Z., Zhang, H.J., Dong, M.K. MAGE: Multi-Agent Environment. ICCNMC-03, 2003
5. Genesereth, M.R., Ketchpel, S.P.. Software Agents. *Comm. of ACM*, 37(7): 48-53.1994.
6. Ricordel, P.M., Demazeau, Y.. From Analysis to Deployment: a Multi-Agent Platform Survey. *Proceedings of the First International Workshop on Engineering Societies in the Agents' World*, Berlin, August 2000.
7. Martin, D., Cheyer, A. Moran, D.. The Open Agent Architecture: A Framework for Building Distributed Software Systems. *Applied Artificial Intelligence*, 13(1-2):92-128, 1999.
8. Sycara, K. Decker, K., Pannu, A., Williamson, M., Zeng, D.. Distributed intelligent agents. *IEEE Expert*, 11(6), 1996.
9. Deb, K., Pratab, A., Agarwal, S., MeyArivan, T.. A fast and Elitist multiobjective genetic algorithm: NSGA-II. *IEEE Trans. Evol. Comput.*, vol.6, pp. 182-197, Apr.2002.
10. Socha, K., Dorohinicki, M.K.. Agent-based Evolutionary Multiobjective Optimisation. *Proceedings of IEEE Conference on Evolutionary computation 2002*.
11. Naithoh, K., Terano, T.. Agent-based Modeling of Corporate Behaviors with Evolutionary Computation. *Proceedings 2003 IEEE International Symposium on Computational Intelligence in Robotics and Automation 2003*, Japan.
12. Vacher. J.P., Galinho. T., Lesage. F., Cardon, A.. Genetic algorithms in a multi-agent system, *IEEE International Joint Symposia on Intelligence and Systems*, 1998, 17-26.
13. Cetnarowicz, K., Kisiel, D.M., Nawarecki, E. The application of evolution Process in Multi-agent World to the Prediction system. *Proceeding of the 2<sup>nd</sup> International Conference on Multi-Agent Systems*.

# Model for Negotiating Prices and Due Dates with Suppliers in Make-to-Order Supply Chains

Lanshun Nie, Xiaofei Xu, and Dechen Zhan

School of Computer Science and Technology,  
Harbin Institute of Technology, Harbin 150001, P.R. China  
{nls, xiaofei, dechen}@hit.edu.cn

**Abstract.** This study deals with the problem of supporting negotiation on prices and due dates of multiple orders between a manufacturer and its suppliers in a make-to-order supply chain. A new negotiation agenda with two phases is proposed based on the fact that relationship between the partners is both cooperative and competitive. In the cooperative phase a mediator implementing Simulated Annealing is incorporated to help the manufacturer and the supplier search tentative agreement of due dates which minimizes the total supply chain cost. Then, they adjust the reservation value and aspiration value of price accordingly based on the idea of integrated-utility. They bargain on the price issue using concession based methods in the competitive phase. The proposed negotiation agenda and approach can achieves near-optimal social welfare and reach win-win solution for negotiation agents. Result of a numerical example is reported.

## 1 Introduction

Pressed by market globalization and concomitant competition, more and more manufacturers are relying on their suppliers to provide raw materials and component parts so as to focus on their core competence. Make-to-order (MTO) or build-to-order supply chains have been constructed to meet various customer requirement while keeping flexible and responsive [1]. The role of the suppliers and the coordination with them are critical in these environments, since the flow of materials is triggered by dynamic customer orders and there is little work-in-process to buffer coordination inefficiencies. However it is natural for each party in the supply chain to try to maximize its individual benefit since they are rational and self-interested. From the manufacturer's point of view, this means getting the materials delivered within a preferred time and at a low price. From the supplier's point of view, this means trying to obtain high prices for materials and/or long delivery times. Thus effective coordination between them is difficult to achieve. The manufacturer and its suppliers have to negotiate on the price and due date of orders to make an agreement and coordinate the operation activities. For both manufacturer and its suppliers, the negotiation process is tightly integrated to their production planning decisions, with price and due date of orders being the main dimensions on which negotiation occurs.

The negotiation process is very complex since it has multiple negotiation issues and is tightly coupled with production planning. If relying upon human only, the process will be not only time-consuming and costly, but also lose the opportunity to maximize social welfare and reach ‘win-win’, then ‘leave the money on the table’ [2]. As a result, the question arises of how to develop appropriate model and system to support negotiating price and due date of orders, coordinating their operation activities and helping them to achieve global efficiency and high local profit.

Several researches have developed models for due-date negotiation between a MTO company and its customers. Wang *et al* have applied a fuzzy-based model to the due-date bargaining process for a MTO company [3]. El-Hafsi and Roland have developed a model for negotiating price and due date in which the probabilities of machine failures are considered [4]. Moodie proposed a simulation model to evaluate various strategies for negotiating price and due date in terms of several operational performance indicators [5]. On the other hand, limited researches have been focus on the models and techniques for negotiating with suppliers [6]. Calosso *et al* studied a similar problem to ours and suggested that it would be appropriate to develop a negotiation support system with an intervenor [2]. Their method has to exhaustively enumerate possible agreements in the negotiation state space and compute their utility, which is infeasible for intractable problem. Cakravastia *et al* have developed a model that can be used to support the process of negotiation between a manufacturer and its multiple suppliers, where each supplier supplies a different part, in a MTO environment [7]. They further developed an integrated model for supplier selection and negotiation in a MTO environment [8]. Their models generate a set of effective alternatives for manufacturer in each round of the negotiation process.

This study deals with the problem of supporting negotiation on the price and due date of multiple orders between a manufacturer and its supplier in a MTO supply chain. We develop model of the decision problems the manufacturer and the suppliers face in the negotiation. Then, an agenda with two phases is proposed based on the fact that relationship between the partners is both cooperative and competitive. This agenda appropriately incorporates both the joint gain searching method and the concession based method into a negotiation framework. Adjustment on the reservation value and aspiration value of price issues based on the idea of integrated utility connects these two phases. The proposed negotiation agenda and approach can achieves near-optimal social welfare and ‘win-win’ contracts. More attention has been paid to computational models of the negotiation framework, which allows agents to find solutions in intractable multi-optima negotiation spaces. The proposed model and method are demonstrated by a numerical example.

## 2 Negotiation Issues and Negotiation Decision Models

Considering a make-to-order supply chain with three tiers. On the customer-level there are multiple orders with specific due date. The manufacturer provides products for these customers. It also procures components from some suppliers



for its manufacturing and assembling products. The marketing department of the manufacturer is responsible for responding to customer inquiries, preparing quotations of price and due date for an order, and conducting negotiations with customers. After the agreement with customers has been made, the production schedule for the orders is generated. The schedules objective is to achieve delivery by the promised dates at the minimum production costs. Based on this production schedule, one determines what components should be purchased and when they will be required. Then the manufacturer negotiates with suppliers and hopes to procure the components at the minimum price while ensuring availability. On receiving the requirements of components from the manufacturer, the supplier try to generate its production schedule to meet the due dates at the minimum production costs. The supplier negotiates with the manufacturer to obtain high order prices. We study the problem of one manufacturer negotiating price and due date of multiple orders with one supplier. Decision models the manufacturer and the supplier face in the negotiation process are developed.

### 2.1 Negotiation Decision Model for the Manufacturer

To complete the orders from customers, assembly jobs and in-house part fabrications take place in the manufacturer. Also, some components needed by the assembly jobs are procured from the supplier. The manufacturer’s total cost is composed of in-house manufacturing cost and external procurement cost. The negotiation decision model for the manufacturer here is modified from the model of Kolish [9].

There are  $A \geq 0 (a=1,2,\dots,A)$  different customer orders, where order  $a$  has the due date  $d_a \geq 0$ . Order  $a$  is made of assembly jobs  $j=s_a,\dots,s_e$ . The number of all assembly jobs is  $J=e_A$ . The manufacturer has  $R^A \geq 0 (r=1,2,\dots,R^A)$  different types of assembly resources. Assembly resource type  $r$  has an available capacity of  $C_{r,t}^A \geq 0$  units at time instant  $t \geq 0$ . Assembly job  $j$  has a processing time of  $p_j \geq 0$  and require  $c_{j,t} \geq 0$  units of assembly resource  $r$ . The assembly jobs are interrelated by an assembly network where job  $j$  has a set  $\rho_j$  of immediate predecessor jobs. The sum of the holding cost for all parts which are assembled by job  $j$  equals  $h_j^A$  per period. The earliest start time of job  $j$  is  $ES_j$  and the latest start time is  $LS_j$ . The manufacturer fabricates  $I \geq 0 (i=1,2,\dots,I)$  types of parts in-house. The number of part units of type  $i$  needed by assembly job  $j$  is  $q_{j,i} \geq 0$ . The in-house fabrication comprises  $R^F$  different production resources. Fabrication resource  $r=1,2,\dots,R^F$  in period  $t$  has  $C_{r,t}^F$  capacity. Each part type  $i$  is solely fabricated by resource  $r_i^F \in \{1,\dots,R^F\}$  in a single level production process. For the production of one unit of part  $i$ , capacity  $c_i$  is needed. Fixed setup cost of  $s_i$  is incurred in each period part  $i$  is produced. The holding cost of part  $i$  is  $h_i^F$ . The manufacturer procures  $K (k = 1, 2, \dots, K)$  types of part from the supplier, where part  $k$  has the due date  $pd_k$  and price  $pr_k$ . The set of external parts needed by assembly job  $j$  is denoted as  $K_j$ . We assume these parts are purchased for orders and have no inventory buffer. Without loss of generality, we assume the number of part  $k \in K_j$  needed by assembly job  $j$  is 1. The decision variables are as follows: The assembly job start flag  $x_{j,t}$  equals to 1 if job  $j$  is

started at time instant  $t$ , and 0 otherwise. The quantity of part type  $i$  produced in period  $t$  is  $Q_{i,t}$ . The quantity of part type  $i$  which is in inventory at the end of period  $t$  is  $I_{i,t}$ . The setup variable  $y_{i,t}$  equals to 1 if production for part type  $i$  in period  $t$  takes place, and 0 otherwise.

$$\min ZM_1 = \sum_{a=1}^A \sum_{j=s_a}^{e_a} h_j^A (d_a + 1 - \sum_{t=ES_j}^{LS_j} x_{j,t}) + \sum_{i=1}^I \sum_{t=0}^T (h_i^F I_{i,t} + s_i y_{i,t}) \quad (1)$$

$$\min ZM_2 = \sum_{k=1}^K pr_k \quad (2)$$

s.t.

$$\sum_{t=ES_j}^{LS_j} (t + p_{e_a}) x_{e_a,t} \leq d_a, (a = 1, \dots, A) \quad (3)$$

$$\sum_{t=ES_j}^{LS_j} x_{j,t} = 1, (j = 1, \dots, J) \quad (4)$$

$$\sum_{t=ES_h}^{LS_h} (t + p_h) x_{h,t} - \sum_{t=ES_j}^{LS_j} t x_{j,t} \leq -t_{h,j}^{min}, (j = 1, \dots, J; h \in \rho_j) \quad (5)$$

$$\sum_{j=1}^J \sum_{t=max\{0, t-p_j\}}^{t-1} c_{j,r} x_{j,t} \leq C_{r,t}^A, (r = 1, \dots, R^A; t = 1, \dots, T) \quad (6)$$

$$I_{i,t-1} + Q_{i,t} - \sum_{j=1}^J q_{j,i} x_{j,t} = I_{i,t}, (i = 1, \dots, I; t = 1, \dots, T) \quad (7)$$

$$\sum_{i=1|r_i^F=r}^I c_i Q_{i,t} \leq C_{r,t}^F, (r = 1, \dots, R^F; t = 1, \dots, T) \quad (8)$$

$$y_{i,t} \sum_{j=1}^J q_{j,i} \geq Q_{i,t}, (j = 1, \dots, I; t = 1, \dots, T) \quad (9)$$

$$\sum_{t=1}^{pd_k} x_{j,t} = 0, (j = 1, \dots, J; k \in K_j) \quad (10)$$

$$x_{j,t} \in \{0, 1\}, (j = 1, \dots, J; t = 1, \dots, T) \quad (11)$$

$$y_{j,t} \in \{0, 1\}, Q_{i,t} \geq 0, I_{i,t} \geq 0, (i = 1, \dots, I; t = 1, \dots, T) \quad (12)$$

The objective (1) minimizes total in-house production cost. The objective (2) means the manufacturer tries to minimize total procurement costs by bargaining. Constraints (3) assures that each order is finished not later than its due date. (4) secures that the final assembly can not be delivered to the customer before each assembly job has been executed exactly once. The procedure constraints is given by (5). (6) models the capacity constraint of the assembly resources. (7) links the part flow between the fabrication and the assembly. The fabrication capacity constraints are given in (8). The production and setup relationship is represented by constraints (9). Constraints (10) enforces that the assembly job can start only when the procurement parts it needs are available.

### 2.2 Negotiation Decision Model for the Supplier

The supplier provides parts for the manufacturer. The objective of it is profit maximization, i.e., obtaining higher order price and minimizing total production cost by optimal production planning.

There are  $T(t = 1, \dots, T)$  periods in planning horizon. The supplier has  $M(i = 1, \dots, M)$  order, of which  $K$  orders are from the manufacturer. The due date of order  $i$  is  $d_i$  and the price is  $p_i$ . The supplier has  $N(j = 1, \dots, N)$  different types of resources. The total quantity of resource  $j$  demanded by order  $i$  is  $r_{ij}$ . The unit cost of resource  $j$  in regular work is  $CRG_j$ , while is  $COV_j$  in overtime work. The regular capacity of resource  $j$  in period  $t$  is  $CAPR_{jt}$ , and the overtime capacity is  $CAPO_{jt}$ . The decision variables are as follows: The quantity of resource  $j$  dispatched to order  $i$  in period  $t$  is  $y_{ijt}$ . The planning quantity of resource  $j$  working on overtime state in period  $t$  is  $o_{jt}$ .

$$\min ZS_1 = \sum_{j=1}^N \sum_{t=1}^T CRG_j \left( \sum_{i=1}^M y_{ijt} - o_{jt} \right) + \sum_{j=1}^N \sum_{t=1}^T COV_j o_{jt} \quad (13)$$

$$\max ZS_2 = \sum_{i \in K} p_i \quad (14)$$

s.t.

$$\sum_{t \leq d_i} y_{ijt} \geq r_{ij}, \forall i, j \quad (15)$$

$$\sum_i y_{ijt} \leq CAPR_{jt} + o_{jt}, \forall j, t \quad (16)$$

$$o_{jt} \leq CAPO_{jt}, \forall j, t \quad (17)$$

$$y_{ijt} \geq 0, \forall i, j, t \quad (18)$$

$$o_{jt} \geq 0, \forall j, t \quad (19)$$

The objective (13) minimizes the total production cost of the supplier. The objective (14) means that the supplier bargains with the manufacturer to maximize the total price of the orders. Constraints (15) ensures that the amount of resources allocated to an order is at least equal to the required amount. (16) represents the total capacity available, while (17) represent the capacity limits of overtime work. Constraints (18) and (19) define existence domains of the decision variables.

### 3 Two-Phases Based Negotiation Agenda and Computational Models

Negotiation can be defined as ‘the process by which a joint decision is made by two or more parties. The parties first verbalize contradictory demands and then move towards agreement by a process of concession making or search for new alternatives’ [10]. The research work on negotiation and bargaining has been conducted in the fields of economics, in particular game theory, and artificial intelligence (AI). The research in the economics community focuses on the outcome of a bargaining situation that satisfies some axioms, or the strategy equilibrium of agents, based on some rigorous assumptions. Researchers in the field of AI contribute efforts to develop software agents which should be able to negotiate in an intelligent way on behalf of their users. The negotiation support methods in AI can be further divided into two categories: concession based methods and joint gains seeking methods. In concession based methods the parties start negotiations from separate positions and alternative negotiation protocol is usually adopted. These methods are suitable to competitive and distributive negotiations. In joint gains seeking methods the parties start from a jointly accepted position and mediated single negotiation text protocol is usually adopted. These methods are suitable to cooperative and integrative negotiations [11] [12]. Agenda is critical to multi-issue negotiation. There usually exist three types of negotiation procedure: separate, simultaneous and sequential [13].

The relationship between the manufacturer and the supplier in supply chain is both cooperative and competitive. When considering the goal of coordinating and synchronizing the manufacturing activities, providing satisfying product for final customers and improving the performance of the whole supply chain, both the manufacturer and the supplier are consistent and ‘cooperative’. The objective  $ZM_1$  of the manufacturer and the objective  $ZS_1$  of the supplier reflect this kind of relationship. However, they are competitive or conflicting in the transfer payment from the manufacturer to the supplier, in other word, the division of the profit since they are rational and self-interested. The competitive relationship is shown in the objective  $ZM_2$  of the manufacturer and the objective  $ZS_2$  of the supplier. Because of this combined relationship, neither joint gains seeking methods nor concession based methods can effectively solve the price and due date negotiation problem in MTO supply chains. Then, a new negotiation agenda entitled *SCAG* is proposed in this study. It is composed of two phases: Non-price issues, i.e. due dates, are negotiated cooperatively in the first phase. A mediator agent

is incorporated into the negotiation framework and joint gains seeking method is used to help the manufacturer and the supplier search the agreement point with lowest cost of the whole supply chain. The price issues are negotiated in the second phase. They are competitive and adopt concession based methods to reach agreement. The two phases are connected by the utility exchange ensuring integrated utility of multiple issues.

### 3.1 Negotiating the Due Dates

Because of group rationality and the cooperative nature on the due date issues, a mediator is incorporated into this phase and single negotiation text protocol is adopted. The mediator agent proposes a solution, then the manufacturer agent and the supplier agent evaluate this solution based on their negotiation decision models. This process continues until joint gains can not be improved. The utility of due date issues are complex nonlinear function since the negotiation decision model are linear programming or integer programming. Thus the mediator should have great capability in searching negotiation state space. Simulated annealing has proven effective in optimization because it can travel through utility valleys on the way to higher optima [14]. So this study implements a mediator with simulated annealing searching algorithm. We denote the manufacturer agent as MA, the supplier agent as SA and the mediator agent as IA from now [15].

The first phase of the negotiation is outlined as follows:

**Step 1.** MA makes production planning decision without objective  $ZM_2$  and constraints (10). The optimal production cost is  $ZM_{1,0}$  and the requirement dates of the procurement parts are  $pd_{k,0}$  ( $k = 1, 2, \dots, K$ ). Given this, MA determines its reservation value  $pr_{mr,0}$  and aspiration value  $pr_{ma,0}$  of the total procurement price. Then MA sends the due dates requirement to SA and IA, and they start due date negotiation from this point. Also MA sends its evaluation for this solution, i.e.  $ZM_{1,0}$ , to IA.

**Step 2.** SA makes production planning decision without the objective  $ZS_2$  after receiving the due date requirement from MA and gets the optimal production cost  $ZS_{1,0}$ . Given this, SA determines its reservation value  $pr_{sr,0}$  and aspiration value  $pr_{sa,0}$  of the total order price. SA sends its evaluation for this solution, i.e.  $ZS_{1,0}$ , to IA.

**Step 3.** IA initializes the parameters of simulated annealing algorithm. The initial solution  $x_i := pd_{k,0}$ . The objective  $f(x_i) := ZM_{1,0} + ZS_{1,0}$ .  $s := 0$ . Initial temperature  $t_0 = t_{max}$ .

**Step 4.** IA randomly selects a solution  $x_j$  from the neighborhood region  $N(x_i)$  of  $x_i$  and sends it to both MA and SA for evaluation.

**Step 5.** MA makes production planning decision considering the constraints (10) with the parameters  $x_j$ , computes optimal production cost  $ZM_{1,j}$  and sends it to IA.

**Step 6.** SA make production planning decision with the due date of orders  $x_j$ , computes optimal production cost  $ZS_{1,j}$  and sends it to IA.

**Step 7.** IA computes  $\Delta f_{ij} = f(x_j) - f(x_i)$  where  $f(x_j) = ZM_{1,j} + ZS_{1,j}$ . If  $\Delta f_{ij} \leq 0$ , set  $x_i := x_j$ , else if  $\exp(-\Delta f_{ij}/t_s) > \text{random}(0, 1)$ , set  $x_i := x_j$ . If the termination criteria on this temperature is met, goto **Step 8**, else goto **Step 4**.

**Step 8.** IA decreases the temperature  $t_{s+1} := d(t_s)$ .  $s := s + 1$ . If the termination criteria of the searching procedure is met, IA sends the optimal solution of order due dates  $x_{opt}$  to MA and SA, goto next two steps, else goto **Step 4**.

**Step 9.** MA adjusts its reservation value and aspiration value of the price according to the change of utility on due date issues(cost) caused by the result of previous negotiation. Denoting the MA's optimal production cost constrained by  $x_{opt}$  as  $ZM_{1,e}$ , now the reservation value is  $pr_{mr,e} := pr_{mr,0} + ZM_{1,0} - ZM_{1,e} - ESTC_m$  and aspiration value is  $pr_{ma,e} := pr_{ma,0} + ZM_{1,0} - ZM_{1,e} - ESTC_m$ , where  $ESTC_m$  is the estimation of benefit MA should obtain from the decrease of total supply chain cost.

**Step 10.** SA adjusts its reservation value and aspiration value of the price according to the change of utility on due date issues(cost) caused by the result of previous negotiation. Denoting the SA's optimal production cost constrained by  $x_{opt}$  as  $ZS_{1,e}$ , now the reservation value is  $pr_{sr,e} := pr_{sr,0} - ZS_{1,0} + ZS_{1,e} + ESTC_s$  and aspiration value is  $pr_{sa,e} := pr_{sa,0} - ZS_{1,0} + ZM_{1,e} + ESTC_s$ , where  $ESTC_s$  is the estimation of benefit SA should obtain from the decrease of total supply chain cost.

Near-optimal due date solution and social welfare are achieved through the negotiation of this phase. Also, the acceptable regions of price issues for both the manufacturer and the supplier have been adjusted based on the negotiation result of this phase and the idea of integrated utility, that is, utilities of different interdependent issues can be exchanged while keeping equivalence of total utility [16]. This makes it easy and possible that the manufacturer and the supplier reach win-win solution on the price issue in the next negotiation phase.

This study makes an assumption that MA and SA are willing to report their production cost associated with due date solution to IA for the purpose of maximizing social welfare. This is reasonable and feasible because: (a) The manufacturer and the supplier are cooperative in the due date issues; (b) The behavior of IA is unbiased and admitted by both the manufacturer and the supplier before the negotiation starts; (c) Changes on their production cost influence their acceptable regions of price issue. Temporal decrease in the utility can be compensated in the next phase; (d) The result of this phase will not be implemented until they reach agreement on price issues and they still have chance to bargain or even terminate with their BATNA(best alternatives to a negotiated agreement) at this moment.

### 3.2 Negotiating the Price

Because of individual rationality and the competitive nature on the price issues, alternative offers protocol and concession based methods are adopted in this phase.  $[pr_{ma,e}, pr_{mr,e}]$  ( $[pr_{sr,e}, pr_{sa,e}]$ ) is the range of values for price that are acceptable to MA(SA). The agents alternatively propose offers at times in

$\tau = \{0, 1, \dots\}$ . Each agent has a deadline.  $T^m(T^s)$  denotes deadline of MA(SA). Let  $p_{m \rightarrow s}^t$  ( $p_{s \rightarrow m}^t$ ) denote the price offered by MA(SA) at time  $t$ . The agent who makes the initial offer is selected randomly at the beginning of this phase. On receiving an offer from MA at time  $t$ , the action taken by SA in response is as follows (The action taken by MA is similar on receiving an offer from SA.):

- (1) If the offer is termination offer, SA terminates the negotiation.
- (2) SA rates the offer using its utility function  $U^s$ . If the value of  $U^s$  for  $p_{m \rightarrow s}^t$  at time  $t$  is greater than the value of the counter-offer SA is ready to send in the next time period,  $t'$ , i.e.,  $U^s(p_{m \rightarrow s}^t, t) \geq U^s(p_{s \rightarrow m}^t, t')$  for  $t' = t + 1$ , then SA accepts the offer at time  $t$ . Negotiation ends successfully in an agreement. Otherwise it is necessary to continue negotiating.
- (3) If  $t > T^s$ , the deadline is exceeded. SA sends a termination offer to MA and terminates the negotiation.
- (4) Otherwise a counter-offer  $p_{s \rightarrow m}^{t'}$  is made in the next time period  $t'$ .

Many researches have been focus on the bilateral price negotiation problem [17] [18]. We simply present a model under time constraints and in an incomplete information setting, which is similar to Fatima [17]. Agents' utilities of price issues are defined with the following utility functions which incorporate the effect of time discounting:

$$\begin{aligned} \text{MA: } U^m(p, t) &= U_p^m(p)U_t^m(t), U_p^m(p) = pr_{mr,e} - p, U_t^m(t) = (\delta^m)^t. \\ \text{SA: } U^s(p, t) &= U_p^s(p)U_t^s(t), U_p^s(p) = p - pr_{sr,e}, U_t^s(t) = (\delta^s)^t. \end{aligned}$$

$\delta^m$  and  $\delta^s$  are the discounting factor.

Agents' counter-offer generation are defined as follows:

$$\begin{aligned} \text{MA: } p_{m \rightarrow s}^t &= pr_{ma,e} + \phi^m(t)(pr_{mr,e} - pr_{ma,e}). \\ \text{SA: } p_{s \rightarrow m}^t &= pr_{sr,e} + (1 - \phi^s(t))(pr_{sa,e} - pr_{sr,e}). \end{aligned}$$

Function  $\phi^m(t)$  and  $\phi^s(t)$  are the negotiation decision function [19] and defined as follows:

$$\phi^a(t) = k^a + (1 - k^a)\left(\frac{t}{T_a}\right)^{1/\psi}, a \in \{m, s\} \tag{20}$$

the value of  $\psi$  determines agents' speed of concession, such as Boulware, Conceder or Linear. In a word, the value of a counter offer depends on the initial price (aspiration value) at which the agent starts negotiation, the final price (reservation value) beyond which the agent does not concede, the time  $t$  at which it offers the final price, and  $\psi$ . These four variables form an agent's strategy.

Social welfare has been maximized and acceptable regions of price have been adjusted appropriately. So in this phase by carefully defining the utility function and the strategy, agent MA and SA will optimally negotiate with each other on behalf of the manufacturer and the supplier, and reach win-win agreement on price issues.

### 3.3 Some Characteristics of the Negotiation Agenda *SCAG*

Characteristics of the proposed agenda and framework *SCAG* for price/due date negotiation in supply chain are as follows:

(1) Near-optimal social welfare can be achieved quickly by separating the cooperative issues, agents reporting their true utility and the mediator searching the global negotiation state space effectively.

(2) Inter-dependent relationship between cooperative issues and competitive issues are not broken in *SCAG*, so agents don't lose the chance of trade-off between them.

(3) Social welfare maximization and exchange of integrated-utility can improve the probability of successful negotiation.

## 4 Numerical Example

We implement a multi-agent negotiation support system for price/due date negotiation in supply chain based on the negotiation agenda and computational model above, which is composed of manufacturer agent, supplier agent, mediator agent and some adjuvant agents. Agents are implemented in Java on the Aglets platform of IBM. The negotiation decision models are solved using the LINGO 8 standard mathematical programming solver. The following example are solved by this system.

Parameters setting of the manufacturer's decision model:  $A=2$ ,  $d_1=5$ ,  $d_2=3$ .  $s_1=1$ ,  $e_1=3$ ,  $s_2 = e_2=4$ .  $\rho_3=\{1,2\}$ .  $R^A=1$ ,  $C_{1,t}^A=3$ .  $p_1=1$ ,  $p_2=1$ ,  $p_3=1$ ,  $p_4=1$ .  $c_{1,1}=2$ ,  $c_{2,1}=2$ ,  $c_{3,1}=1$ ,  $c_{4,1}=1$ .  $h_1^A=5$ ,  $h_2^A=10$ ,  $h_3^A=5$ ,  $h_4^A=5$ .  $R^F=1$ ,  $C_{1,t}^F=10$ .  $I=3$ .  $c_1=1$ ,  $c_2=1$ ,  $c_3=1$ .  $s_1=20$ ,  $s_2=10$ ,  $s_3=10$ .  $h_1^F=1$ ,  $h_2^F=2$ ,  $h_3^F=1$ .  $q_{11}=5$ ,  $q_{22}=5$ ,  $q_{31}=5$ ,  $q_{43}=5$ .  $K_2=\{1\}$ .  $K_4=\{2\}$ .

Parameters setting of the supplier's decisions model:  $T=5$ .  $M = K=2$ .  $N=2$ .  $r_{11}=15$ ,  $CRG_1=1$ ,  $COV_1=3$ ,  $CAPR_{1t}=10$ ,  $CAPO_{1t}=8$ .  $r_{22}=30$ ,  $CRG_2=1.5$ ,  $COV_2=2.5$ ,  $CAPR_{2t}=20$ ,  $CAPO_{2t}=10$ .

When the negotiation starts, the manufacturer determines requirement dates of procurement orders  $d_1=1$ ,  $d_2=1$ . The manufacturer sets its reservation value  $pr_{mr,0}=100$ , aspiration value  $pr_{ma,0}=80$ ,  $\delta^m=1$ ,  $k^m=0$  and  $\psi^m = 0.9$ . The supplier set its reservation value  $pr_{sr,0}=85$ , aspiration value  $pr_{sa,0}=110$ ,  $\delta^s=1$ ,  $k^s=0$  and  $\psi^s=1.1$ . First we let MA and SA negotiate the price issue directly and denote the solution as *PND*. We use it as a benchmark for evaluating the performance of the *SCAG*. Secondly, MA and SA, supported by IA, negotiate according to the proposed agenda *SCAG*. The negotiation results are shown in Table 1 (TOC–total cost, U\_Ins–utility increase).

As can be seen in Table 1, negotiation in the first phase of agenda *SCAG* reaches the optimal due date solution  $d_1=2$ ,  $d_2=1$ . This decreases the total supply chain in 5 units although the cost of the manufacturer increase temporally. Then by appropriately adjusting the reservation value and aspiration value of price issues and negotiating in the second phase, not only the temporal utility decrease of the manufacturer is compensated, but also the increased social



**Table 1.** The result of negotiation for example problem

	supplier			agreement	manufacturer				
	reservation	aspiration	cost	profit	point	reservation	aspiration	cost	TOC
<i>PND</i>	85	110	80	12.05	(1,1,92.05)	100	80	120	212.05
<i>SCAG</i>	75	100	70	12.31	(2,1,83.31)	92	72	125	208.31
U_Ins				1.26					3.74

welfare is fairly divided. Both the manufacturer’s utility and the supplier’s utility increase compared with *PND*, and the win-win solution is achieved.

## 5 Conclusions

This paper presented a new model for price/due date negotiation between the manufacturer and its suppliers in a make-to-order supply chain. By developing negotiation decision models for them, this study finds that they are cooperative on due date issues, i.e. improving performance of the whole supply chain, while competitive on price issues, i.e. transfer payment from the manufacturer to the supplier. So a new negotiation agenda and framework *SCAG* with two phases is proposed. In the first phase a mediator agent implementing simulated annealing is incorporated to help the manufacturer and the supplier search optimal due date solution and maximize social welfare. We have prove its feasibility by analyzing the behavior of the agents. They negotiate price issues and reach win-win solution in the second phase after reservation values and aspiration values of price issues have been adjusted according to the result of the first phase. A numerical example is provided here to validate this procedure. Operation activities in MTO supply chains can be effectively coordinated and the production planning of both the manufacturer and the suppliers can be optimized without adding undue complexity and effort by the proposed mechanism and system. They are also useful for developing online Business-to-Business trading platforms for electronic commerce.

The system is now applied to aid order negotiation and activity coordination in a supply chain composed of a large air-condition manufacturer and its suppliers in China. Also, the proposed agenda and framework can be extended to multi-issue negotiation in make-to-stock supply chains. In future, we will generalize this framework to more potential application areas and introduce learning to the model.

## References

1. Gunasekarana A., Ngai E. W. T.: Build-to-order supply chain management: a literature review and framework for development. *Journal of Operations Management*. **23** (2005) 421–451
2. Calosso T., Cantamess M., Gualano M.: Negotiation support for make-to-order operations in business-to-business electronic commerce. *Robotics and Computer-Integrated Manufacturing*. **20** (2004) 405–416

3. Wang D., Fang S. C., Hodgson T. J.: A fuzzy due-date bargainer for make-to-order manufacturing system. *IEEE Transactions on Systems, Man, and Cybernetics -Part C, Application and Review*. **28** (1998) 492–497
4. El-Hafsi M., Roland E.: Negotiating price/delivery date in a stochastic manufacturing environment. *IIE Transactions*. **31** (1997) 255–270
5. Moodie D. R.: Demand management: the evaluation of price and due date negotiation strategy using simulation. *Production and Operation Management*. **8** (1999) 151–162
6. Miller P. A., Kelle P.: Quantitative support for buyer-supplier negotiation in just-in-time purchasing. *International Journal of Purchasing and Materials Management*. **Spring** (1998) 25–30
7. Cakravastia A., Nakamura N.: Model for negotiating the price and due date for a single order with multiple suppliers in a make-to-order environment. *International Journal of Production Research*. **40(14)** (2002) 3425–3440
8. Cakravastia A., Takahashi K.: Integrated model for supplier selection and negotiation in a make-to-order environment. *International Journal of Production Research*. **42(21)** (2004) 4457–4474
9. Kolish R.: Integration of assembly and fabrication for make-to-order production. *International Journal of Production Economics*. **68** (2000) 287–306
10. Pruitt D.: *Negotiation Behavior*. Academic Press, 1981.
11. Ehtamo H., Hamalainen R. P.: Iterative multiple-criteria methods for reaching Pareto optimal agreement in negotiation. *Group Decision and Negotiation*. **10** (2001) 475–491
12. Kersten G. E.: Modeling distributive and integrative negotiations. Review and revised characterization. *Group Decision and Negotiation*. **10** (2001) 493–514
13. Indest R.: Multi-issue bargaining with endogenous agenda. *Games and Economic Behavior*. **30** (2000) 64–82
14. Bar-Yam, Y. : *Dynamics of Complex Systems*. Reading, Mass.: Addison-Wesley, xvi. (1997) 848
15. Klein M., Faratin P.: Negotiating complex contracts. *Group Decision and Negotiation*. **12** (2003) 111–125
16. Guo Q., Chen C.: An interated-utility based optimization in multi-issues negotiation. *Journal of Software*. **15(5)** 2004 706–711
17. Fatima S., Wooldridge M., Jennings N. R.: Optimal negotiation strategies for agents with incomplete information. *Intelligent Agents VIII. Agent Theories, Architectures and Languages*. Berlin: Springer. (2002) 377–392
18. Zeng D., Sycara K.: Bayesian learning in negotiation. *Int. J. Human-Computer Studies*. **48** (1998) 125–141
19. Faratin P., Sierra C., Jennings N. R.: Negotiation decision function for autonomous agents. *International Journal of Robotics Autonomous Systems*. **24(3-4)** (1998) 159–182.

# Interest-Based Negotiation as an Extension of Monotonic Bargaining in 3APL

Philippe Pasquier<sup>1</sup>, Frank Dignum<sup>3</sup>, Iyad Rahwan<sup>2,4</sup>, and Liz Sonenberg<sup>1</sup>

<sup>1</sup> University of Melbourne, Australia

<sup>2</sup> British University of Dubai, UAE

<sup>3</sup> Utrecht University, The Netherlands

<sup>4</sup> (Fellow) University of Edinburgh, UK

**Abstract.** Reframing is a sub-type of interest-based negotiation strategy that enhances bargaining by allowing the negotiators to ask for the underlying goal of the negotiation and propose alternative plan(s) which may entail a deal on alternative issues. This paper (i) presents a negotiation protocol that support both alternate offers monotonic bargaining and reframing and (ii) gives a fully computational and reproducible specification of bargaining and reframing capable negotiation agents using the 3APL agent language.

**Keywords:** Conflict resolution and negotiation, Dialog and interaction protocols, Agent programming languages, frameworks, and toolkits, Agent-oriented software engineering, Agent-based electronic commerce.

## 1 Introduction

Argumentation-based negotiation (ABN) [10] has been proposed as a way to enhance classical negotiation dialogues between artificial agents and progress toward human computer negotiation. Interest-based negotiation (IBN) [9] is the subset of ABN that focusses on the argumentation about the interests underlying the negotiation. However, no work has proposed a concrete – computational and fully reproducible<sup>1</sup> – way to implement IBN.

This paper advances the state of the art by addressing the software engineering level of negotiation agents. It shows how to develop software agents capable of bargaining and reframing, i.e. a sub-type of IBN. The proposed negotiation model consists of:

1. *The negotiation strategies:* an agent strategy is a sequence of actions, which in the context of negotiation mainly consists of offers and responses. Section 2 shows how reframing can be used to extend classic bargaining strategies. The set of possible strategies are constrained by the chosen protocol.

---

<sup>1</sup> We refer the interested reader to [11] for an example of discussion and experiments showing that formal models usually failed to be reproducible computational characterization when not expressed directly in a particular programming environment.

2. *The negotiation protocol*: the protocol specifies the rules of encounter between the negotiators. It defines what are the locutions and their possible sequences. Section 3 specifies the proposed protocol for alternate offers negotiation and reframing.
3. *The information state of agents*: the agents' models reify the negotiation strategy and the negotiation protocol thus providing distributed decision making mechanisms suited to MAS. In our case, we use the 3APL agent framework (presented in Section 4) to specify reframing-capable cognitive agents (Section 5). Because 3APL is a declarative environment, those specifications are also the actual code for the agent programs.

Finally, Section 6 concludes by discussing the results associated with this model and its implementation.

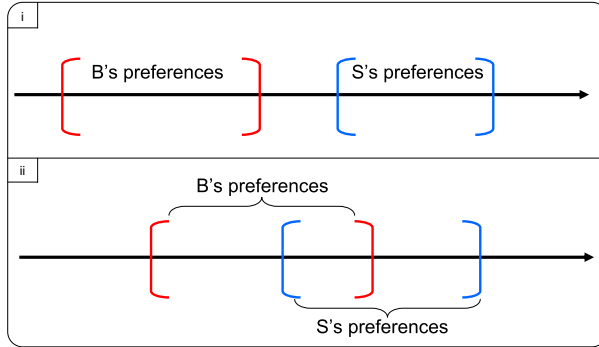
## 2 Negotiation Strategy: From Bargaining to IBN

The challenge of negotiation is to allocate scarce resources to self interested parties. The resources can be quite anything as long as they are scarce, i.e. there is not enough to allow for an allocation that would maximally satisfy all parties. The object of negotiation may be tangible (bandwidth, money, property, processing power, ...) or intangible (better communication, better work performance, more respect, ...). The traditional form of negotiation, characterized by the assertion of opposing positions by the parties, is referred to as position-based negotiation, which in the context of e-business and economic exchanges is addressed as *bargaining*. This tends to view the object of the negotiation as fixed, such that a better deal for one means a lesser beneficial deal for the other; a "zero-sum game".

Bargaining strategies and mechanisms have been well explored both in multi-agent systems and economics [5,8]. One of the properties, which is commonly agreed on in the analysis of bargaining strategies is the monotonicity of the bargainers' offers, i.e. buyers/sellers may only insist on their previous offers or raise/reduce their offers monotonically until an agreement is reached. Time dependant tactics [6], behavior dependent tactics [5] and market driven strategies [12] all have this monotonic offers property. While recent works have been looking at non-monotonic-offers bargaining protocols [14], we will only consider monotonic offers in this paper.

Originally developed for human negotiation and mediation practices and first introduced in [7], the theory of *interest-based negotiation* (IBN) shows that parties are much more likely to come to a mutually satisfactory outcome when the object of the negotiation is not considered as central as the agents' underlying interests. By focusing on interests to be satisfied rather than positions to be won, IBN allows the agents to search the space of negotiation objects (rather than the space of deals for a particular item). When successful, this strategy gives each side more, thereby producing a "win-win" outcome.

IBN has been adapted and applied to MAS negotiation dialogues [9]. In that context, it is a subclass of argumentation-based negotiation (where the agents



**Fig. 1.** Upper and lower bounds on agent preferences ordering

are arguing about negotiation related issues, i.e. beliefs, goals or social aspects). Interest-based negotiation rests on the idea that the agents can explicit the goals underlying the negotiation and discuss alternative ways to achieve these.

While classical negotiation models (heuristic approaches or game theoretic approaches) focus on processes to accommodate the agents' preferences, in IBN, the agents' preferences may change. Let's look at some informal examples to illustrate this idea.

First, consider the following bargaining dialogue where the agents' preferences are summarized by Figure 1, part (i). In that example, the agents fail to reach a deal because it is not possible to accommodate their respective preferences.

*B<sub>1</sub>: I would like to rent a car for 4 days please.*  
*S<sub>1</sub>: I offer you one for \$400.*  
*B<sub>2</sub>: I reject! How about \$200?*  
*S<sub>2</sub>: I reject!*

Of course, sometimes, the agents preferences overlap (as in Figure 1, part (ii)) and a deal can be reached using a monotonic bargaining strategy like in the following dialogue:

*B<sub>1</sub>: I would like to rent a car for 4 days please.*  
*S<sub>1</sub>: I offer you one for \$400.*  
*B<sub>2</sub>: I reject! How about \$200?*  
*S<sub>3</sub>: I reject! How about \$300 then?*  
*B<sub>3</sub>: I guess that's the best I can do, I accept!*

However, it is worth noticing that agents had to make concessions and move away from their preferred outcome in order to reach a deal. The following dialogue gives an example of IBN dialogue where the seller agent (*S*) asks the buyer to give his underlying interest (goal) before making a concession. In this example, the seller knows an alternative way to fulfill the buyer's underlying goal. Assuming that the seller earns a \$100 profit on both the rent of a car for \$400 and a \$200 Quantum ticket, an agreement is reached without any concession.

*B<sub>1</sub>: I would like to rent a car for 4 days please.*

*S<sub>1</sub>: I offer you one for \$400.*

*B<sub>2</sub>: I reject! How about \$200?*

*S<sub>2</sub>: I reject! What do you need a car for?*

*B<sub>3</sub>: I want to drive to Sydney to attend a conference.*

*S<sub>3</sub>: You can also fly to Sydney! I can book you a ticket with Quantum airlines for \$200.*

*S<sub>4</sub>: I didn't know flights were so cheap! I accept!*

This particular negotiation strategy is refereed in the human negotiation literature as *reframing*. The two main strategic issues to model when one want to use reframing in order to enhance bargaining-based negotiation are:

- The bargaining function used by the agent to compute their next offers. While complex bargaining functions are available in the literature, we use a simple but standard one here. Starting with initial offers that are their most preferred options, the agents will use the mean ( $\frac{X+Y}{2}$ ) of the participating agents current offers ( $X$  and  $Y$ ) as long as it falls within the preferences boundaries.
- When to stop bargaining and try reframing? In order to embed IBN in negotiation dialogues, there must be a point where bargaining stops. This point does not have to be the same as when one would only have bargaining available as a strategy. In this paper, reframing is triggered when both agents repeat their offers, which indicates that they reached their less preferred limit and that the bargaining failed.

While the choices made in this paper at these strategic levels are very simple, complex functions could be devised to make the decision based on knowledge of opponent, predictions based on bargaining history, costs of information disclosure, . . . The rest of this paper presents and discusses an implementation of software agents capable of monotonic bargaining and reframing.

### 3 The Negotiation Protocol

In order to enable agents to use reframing, one needs to define an appropriate protocol that allows both bargaining and reframing. As advocated in [13], we break the protocol into two parts, making it more modular. Figure 2 thus presents the UML 2.0 specification of the two parts of the protocol: (1) the bargaining protocol (on the left) and (2) the reframing one (on the right).

The bargaining protocol is a classical alternated offers protocol. It is made of the sequencing of the initiator's request, and the subsequent refusal or proposal from the partner. Then the protocol allows alternated counter proposals from both parties with possible refusal or acceptance at each stage. Original to this protocol is the possibility to embed a reframing dialog instead of making a proposal (or counter proposal).

The reframing protocol mainly consists in allowing its initiator to request the partner for the underlying goal of the negotiation. The partner can inform the

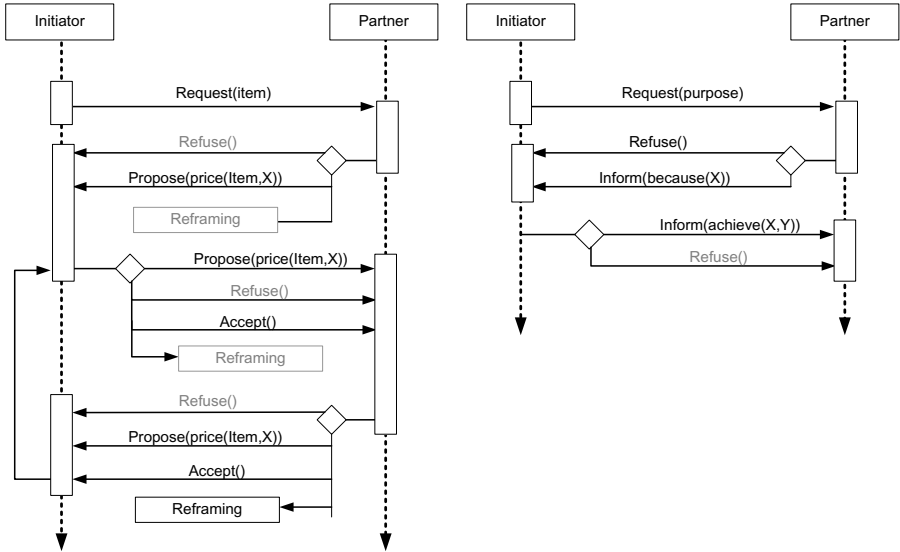


Fig. 2. UML 2.0 specification of the proposed bargaining and reframing protocols

initiator of this or refuse to answer. In the case where the partner informs the bidder of his underlying goal, the initiator can be clueless and quit the protocol with a cancel message or he can inform the partner of an alternative plan for achieving the goal. Reframing, as a strategy, rests on the hypothetic existence of alternative plan(s) to achieve the underlying goal of the initiator involving different resources for which bargaining will be more advantageous for both parties.

#### 4 Agents' Information States: The 3APL Agent Platform

In order to give a fully reproducible computational characterization of the ideas developed here, we will use the 3APL agent language to present our model. Fortunately, the 3APL language has been developed precisely to help bridging the gap between theoretical models (in particular, logic-based) and practical implementation of MAS concepts.

Since 3APL is well documented and discussed in [4,3,2], we only sum up the basics here. A 3APL agent is made of 6 bases:

1. A *beliefs base*: the agent belief base is expressed as a Prolog<sup>2</sup> code (ISO standard).<sup>3</sup> External Prolog programs can be embedded thus allowing to give common knowledge to the agents and outsource more complex reasoning capabilities;

<sup>2</sup> For space reasons, we don't detail the Prolog syntax in here. Suffice to remember that variables are noted with capitalized identifiers.

<sup>3</sup> 3APL use the JIP Prolog java package, see [1] for more information.

```

(Program) ::=
    "Program" <ident>
    ( "Load" <ident> )?
    "Capabilities : " ( <capabilities> )?
    "BeliefBase : " ( <beliefs> )?
    "GoalBase : " ( <goals> )?
    "PlanBase : " ( <plans> )?
    "PG - rules : " ( <p.rules> )?
    "PR - rules : " ( <r.rules> )?

<capabilities> ::= <capability> ( " , " <capability> )*
<capability> ::= "{ " <query> " } " (Atom) "{ " < literals> " } "
<beliefs> ::= ( <belief> )*
<belief> ::= <ground_atom> " . " | <atom> " : - " < literals> " . "
<goals> ::= <goal> ( " , " <goal> )*
<goal> ::= <ground_atom> ( " and " <ground_atom> )*
<plans> ::= <plan> ( " , " <plan> )*
<plan> ::= <basicaction> | <composedplan>
<basicaction> ::= "ε" | (Atom) | "Send"( <iv> , <iv> , <atom> )" " |
    "Java"( <ident> , <atom> , <var> )" " | <wff> "?" | <atom>
<composedplan> ::= "if" <wff> "then" <plan> ( "else" <plan> )? |
    "while" <query> "do" <plan> |
    <plan> " ; " <plan>
<p.rules> ::= <p.rule> ( " , " <p.rule> )*
<p.rule> ::= <atom> " <- " <query> " | " <plan>
<r.rules> ::= <r.rule> ( " , " <r.rule> )*
<r.rule> ::= <plan> " <- " <query> " | " <plan>
<literals> ::= <literal> ( " , " <literal> )*
<literal> ::= <atom> | "not"( <atom> )" "
<wff> ::= <literal> | <wff> "and" <wff> | <wff> "or" <wff>
<query> ::= <wff> | "true"
<iv> ::= <ident> | <var>

```

**Fig. 3.** The EBNF specification of the 3APL language for programming individual agents where:  $\langle atom \rangle$  refers to Prolog ground atomic formulas (can include Prolog-like list representation),  $\langle Atom \rangle$  denotes atomic action formulas (starting with a capital letter),  $\langle ident \rangle$  denotes strings and  $\langle var \rangle$  denotes variables (starting with a capital letter as in Prolog)

2. *A capabilities base:* the capabilities base stores all the mental and external parameterized actions known by the agent. All capabilities are expressed as:  $\{ \langle Precondition \rangle \} \langle Action(parameters) \rangle \{ \langle Effects \rangle \}$ , where  $\langle Precondition \rangle$  is interpreted as a query to the belief base and  $\langle Effects \rangle$  captures the effects of the action  $\langle Action() \rangle$  on the agent's belief base, in terms of addition and deletion of beliefs;
3. *A goal base:* this base contains the agent's goal(s) expressed as grounded Prolog expressions;
4. *A plan base:* the plan base holds the agent's current plans expressed in terms of actions and (sub)plans, using the following planing operators: sequence ( $;$ ), conditional (*if*  $\langle query \rangle$  *then*  $\langle plan \rangle$  *else*  $\langle plan \rangle$ ) and iteration (*While*  $\langle query \rangle$  *do*  $\langle plan \rangle$ );
5. *A goal planning rules base:* the goal planing rules are used to generate plans to achieve goals. These are expressed as:  $\langle head \rangle \leftarrow \langle guard \rangle \mid \langle body \rangle$  where  $\langle head \rangle$  is a goal,  $\langle guard \rangle$  is belief query that checks if the rule can be applied and  $\langle body \rangle$  is the plan to execute;



- 6. A *plan revision base*: the plan revision rules are used to revise plans from the plan base. These are also expressed in terms of head (plan), guard (belief query) and body (plan). The head of the rule is replaced by the body if the guard holds. The rule thus changes the plan and no backtracking is possible;

Furthermore, 3APL includes FIPA compliant communication facilities through the special and built in action  $Send(\langle interlocutor \rangle, \langle performative \rangle, \langle content \rangle)$ . Such action adds the associated  $sent(\langle interlocutor \rangle, \langle performative \rangle, \langle content \rangle)$  and  $received(\langle sender \rangle, \langle performative \rangle, \langle content \rangle)$  beliefs in the interlocutor and sender belief bases respectively.

The EBNF specification of the 3APL language is given by Figure 3. All those structures are used and manipulated through the deliberation cycle of 3APL agents which is illustrated in Figure 4. The 3APL agent platform is made of an interface that allows defining and executing 3APL agents. Facilities like an agent management system (AMS) and a client server distribution are built-in.<sup>4</sup>

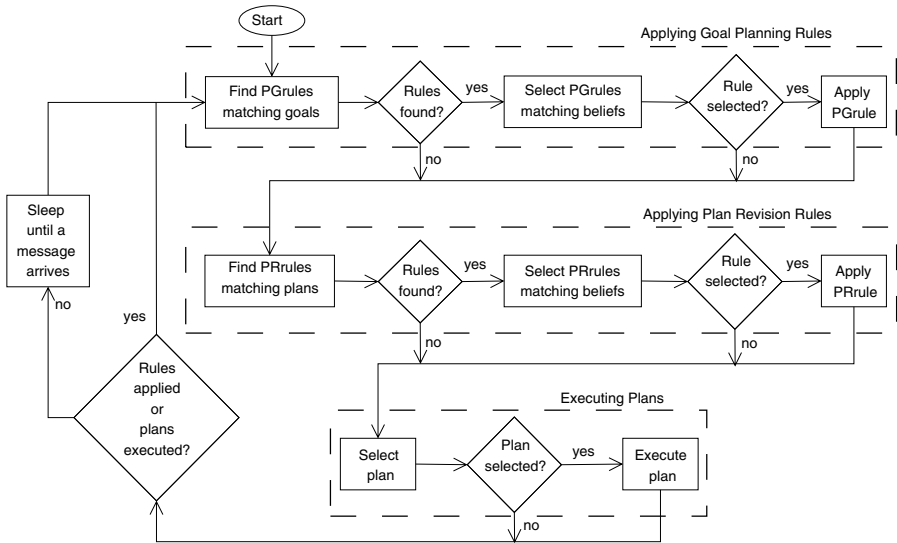


Fig. 4. 3APL agent deliberation cycle

## 5 Implementing Reframing in 3APL: Detailed Example

In this Section, we give the specification-code of generic buyer and seller agents through a concrete example that can easily be adapted to a great number of actual negotiation contexts (e.g. in electronic commerce). Let us consider two agents Vero and Philippe whose specification-codes are indicated by Figures 5 and 6 respectively.

<sup>4</sup> The 3APL distribution is available for download at: <http://www.cs.uu.nl/3apl/>

**PROGRAM "Vero"**


---

```

1: CAPABILITIES {
2: {True} Goal(Y) {goal(Y)}
3: {True} Treatedgo(X) {go(X)}
4: {True} Treated(X,C,Y) {not received(X,propose,content(price(C,Y)))}
5: {True} Updateoffer(Z,W,N) {not currentoffer(Z), not lastoffer(W), lastoffer(Z),
  currentoffer(N)}
6: {True} Updateotheroffer(X,Y) {not otheroffer(X), otheroffer(Y)}
7: {True} Updatecheaper(Y,V) {not cheaper(V,Y), cheaper(Y,V)}
8: {True} Pay(X,Y) {payed(X,Y)}
9: }
10: BELIEFBASE {
11: transport(car). transport(plane). transport(boat). transport(train).
12: conf(aaai). conf(primus). conf(uumus).
13: town(sydney). town(candera). town(melbourne).
14: isin(aaai,sydney). isin(primus, candera). isin(uumus, melbounre)
15: pricereg(aaai,125). pricereg(primus, 75). pricereg(uumus, 200).
16: seller(philippe). seller(sally). seller(bob).
17: achieve(go(Z),W):- town(Z), transport(W).
18: cheaper(rentcar,plane). cheaper(rentcar,train). cheaper(rentcar,boat).
19: cheapest(X):- not cheaper(Y,X).
20: deal(price(C,Z)):- sent(X,accept,content(price(C,Z))), seller(X).
21: deal(price(C,Z)):- received(X,accept,content(price(C,Z))), seller(X).
22: securetransport(X):- deal(price(Y,Z)), transport(Y).
23: prefmin(rentcar,250). prefmax(rentcar,100).
24: prefmin(plane,250). prefmax(plane,150).
25: nextoffer(N):- currentoffer(Z), otheroffer(O), plus(Z,O,P), times(P,0.5,N).
26: currentoffer(0). lastoffer(0). otheroffer(0).
27: }
28: GOALBASE {
29: go(aaai) }
30: PLANBASE {}
31: PG-RULES {
32: go(X)← conf(X) and isin(X,Y) | {Goal(go(Y)); goal(go(Y)); payreg(X); Treatedgo(X)},
33: }
34: PR-RULES {
35: goal(X)← achieve(X,Z) and cheapest(Z) | negotiate(Z)
36: payreg(X)← pricereg(X,Y) and securetransport(X) | {Pay(X,Y)} }
37: negotiate(Z) ← seller(X) | {Send(X,request,content(Z)); bargain(X,Z)}
38: bargain(X,Content) ← received(X,propose,content(price(Content,Y))) and prefmin(Content,M)
  and prefmax(Content,P) and otheroffer(T) and currentoffer(C) | {Treated(X,Content,Y);
  Updateotheroffer(T,Y); minus(Y,C,D)?; if D < 10 then
  Send(X,accept,content(price(Content,Y))) else {nextoffer(X,Content); bargain(X,Content)}},
39: bargain(X,Y) ← received(X,reqpurpose,why(Y)) and achieve(Z,Y) and goal(Z) and not
  sent(X,inform,because(Z)) | {Send(X,inform,because(Z));reframe(Z,Y)},
40: bargain(X,Y) ← received(X,reqpurpose,why(Y)) and achieve(Z,Y) and goal(Z) and
  sent(X,inform,because(Z)) | {Send(X,refuse,because(Z))},
41: bargain(X,Y) ← received(X,accept,content(price(Y,Z))) | {Treated(X,Y,Z)},
42: reframe(Z,V) ← received(X,inform,content(achieve(Z,Y))) and goal(Z) and not cheaper(Y,V) |
  {Updatecheaper(Y,V); goal(Z)},
43: nextoffer(Y,X)← prefmin(X,M) and prefmax(X,A) and currentoffer(Z) and lastoffer(W) | if not
  sent(Y,propose,content(price(X,A))) then {Send(Y,propose,content(price(X,A)))};
  Updateoffer(Z,W,A)} else nextoffer(N)?; if N > M then {Send(Y,propose,content(price(X,M)))};
  Updateoffer(Z,W,M) else {Send(Y,propose,content(price(X,N)))}; Updateoffer(Z,W,N)} }
44: }

```

---

**Fig. 5.** 3APL buyer program

Vero has a number of beliefs about transport means (line 11), conferences (line 12), towns (line 13), conferences locations (line 14), conferences prices (line 15) and seller agents (line 16) (for space reason, we gave this agent knowledge about seller agents rather than having her discover them using 3APL's AMS).

She believes she can go to a town using any of the transport means (line 17) and that a deal is made whenever she or the seller accepts an offer (lines 20 and 21). She believes that she succeeds in securing some transportation when she has made a deal on a particular transport means (line 22). She has preferences' boundaries over the prices of different items (lines 23 and 24). Vero also has a priori beliefs about the relative price of transport means (line 18) and knows that the cheapest would be the one(s) that do not have cheaper alternatives (line 19). Finally, the predicate *nextoffer()* captures the bargaining strategy function (line 25, as described in Section 2) and various variables are initialized (line 26).

Vero's initial goal is to attend to a conference named *aaai* (line 29). She knows that in order to go to a conference she needs to go to the city where the conference is held and pay the registration (line 32). In order to pay the registration, she needs to be aware of the price of the registration and make sure that she has already secured transportation to the place where the conference occurs (line 36). Vero also knows that in order to achieve a goal she will have to negotiate on the cheapest way to accomplish it (line 35). Negotiation starts by sending a request for the wanted item to a seller and then bargain (line 37).

Bargaining is defined recursively using plan reasoning rules. On the reception of a proposal, she accepts it if it is sufficiently close to her own proposal or else, she makes a counter offer and the bargaining continues (line 38). The bargaining stops when she issues an acceptance message (line 38) or when she receives one (line 41). Her strategy for computing the next offer (line 43) matches the bargaining strategy discussed in Section 2, i.e. starting with the maximally preferred offer and increases monotonically up to the less preferred but still acceptable proposal.

Reframing occurs when she receives a request for purpose, which she answers by refusing it (line 40) or by informing the seller about her purpose (lines 39) and prepares for a reframing. If the partner informs her of a valuable alternative to achieve her underlying goal, she will adopt it as a subgoal and will start a new bargaining on it (line 42).

Philippe is a seller agent that maintains prices preferences' boundaries upon the different items he is selling (lines 7 and 8, in Figure 6). His beliefs include the predicate *nextoffer* (line 10) which encode its bargaining strategy and *reframecondition* (line 11) which gives the condition for attempting a reframing (as described in Section 2). The protocol described in Section 3 is implemented in a generic way through 6 rules<sup>5</sup>:

1. when he receives a request for an item that has not been treated, he makes an initial proposal (his most preferred price) and enters a bargaining plan (line 17);
2. on the reception of a counter proposal, he can accept it if it is close enough to its current offer or make another offer (line 20);

---

<sup>5</sup> Note that the parts in grey in Figure 2 are not shown in the agents code for space reason.

**PROGRAM "Philippe"**


---

```

1: CAPABILITIES {
2: {True} Treated(X,C,Y) { not received(X,propose,content(price(C,Y)))}
3: {True} Updateotheroffer(Z,X,Y) {not lastotheroffer(Z), lastotheroffer(X), not otheroffer(X),
  otheroffer(Y)}
4: {True} Updateoffer(X,Y,Z) {not currentoffer(X), not lastoffer(Y), lastoffer(X), currentoffer(Z)}
5: }
6: BELIEFBASE {
7: prefmax(rentcar,500). prefmin(rentcar,300).
8: prefmax(plane,400). prefmin(plane,200).
9: currentoffer(0). otheroffer(0). lastoffer(0). lastotheroffer(0).
10: nextoffer(N):- currentoffer(Z), otheroffer(O), plus(Z,O,P), times(P,0.5,N).
11: reframecondition(X) :- currentoffer(X), lastoffer(X), otheroffer(Z), lastotheroffer(Z).
12: achieve(plane,go(sydney)).
13: }
14: GOALBASE { }
15: PLANBASE { }
16: PG-RULES {
17: ← received(X,request,content(Content)) and not sent(X,propose,content(price(Content,W)))
  and prefmax(Content,Y) and currentoffer(W) and lastoffer(L) | {Updateoffer(L,W,Y);
  Send(X,propose,content(price(Content,Y))); bargain(X,Content)}
18: }
19: PR-RULES {
20: bargain(X,Content) ← received(X,request,content(Content)) and
  received(X,propose,content(price(Content,Y))) and otheroffer(W) and currentoffer(Z) and
  lastotheroffer(P) | {Treated(X, Content, Y); Updateotheroffer(P,W,Y); minus(Z,Y,R)?; If R <
  10 then Send(X,accept,content(price(Content,Y))) else nextoffer(X,Content)},
21: bargain(X,Content) ← received(X,accept,content(price(Content,Y))) |
  {Treated(X,Content,Y)},
22: reframe(X,Content) ← not sent(X,reqpurpose,why(Content)) |
  {Send(X,reqpurpose,why(Content)); reframe(X,Content)} ,
23: reframe(X,Content) ← received(X,inform,because(Y)) and achieve(P,Y) and not
  sent(X,inform,content(achieve(Y,P))) | {Send(X,inform,content(achieve(Y,P)))} ,
24: nextoffer(X,Request)← prefmin(Request,M) and currentoffer(Z) and lastoffer(W) |
  {nextoffer(N)?; if N > M then {Updateoffer(Z,W,N);
  Send(X,propose,content(price(Request,N))); bargain(X,Request)} else { Updateoffer(Z,W,M);
  if not reframecondition(M) then {Send(X,propose,content(price(Request,M))};
  bargain(X,Request)} else reframe(X,Request) } } }
25: }

```

---

**Fig. 6.** 3APL seller program

3. the bargaining succeeds if he send an acceptance or if he receives ones (lines 20 and 21);
4. when he wants to make another offer, Philippe calculates his next proposal and checks if the reframing condition holds to decide whether to continue the bargaining or enter a reframing strategy (line 24);
5. reframing consists in asking the buyer for his underlying purpose (line 22);
6. if informed by the buyer of her underlying goal, reframing also involves sending him information about an alternative mean to achieve that goal, if any (line 23).

In this context, Vero's initial deliberation and planning on how to achieve her goal ends up in a request for a proposal about the rent of a car from one of the sellers she knows (Philippe is the first in her list). The agents bargain until Philippe asks for the underlying purpose of the initial request. Vero informs him that her goal is to go to Sydney. Philippe then informs her of an alternative way to achieve that goal, namely by taking a plane. Notice that because the goal base

Sender	Receiver	Performative	Content
vero	philippe	request	content(rentcar)
philippe	vero	propose	content(price(rentcar,500))
vero	philippe	propose	content(price(rentcar,100))
philippe	vero	propose	content(price(rentcar,300))
vero	philippe	propose	content(price(rentcar,200))
philippe	vero	propose	content(price(rentcar,300))
vero	philippe	propose	content(price(rentcar,250))
philippe	vero	propose	content(price(rentcar,300))
vero	philippe	propose	content(price(rentcar,250))
philippe	vero	reqpurpose	why(rentcar)
vero	philippe	inform	because(go(sydney))
philippe	vero	inform	content(achieve(go(sydney),plane))
vero	philippe	request	content(plane)
philippe	vero	propose	content(price(plane,400))
vero	philippe	propose	content(price(plane,150))
philippe	vero	propose	content(price(plane,275))
vero	philippe	propose	content(price(plane,212.5))
philippe	vero	propose	content(price(plane,243.75))
vero	philippe	propose	content(price(plane,228.125))
philippe	vero	propose	content(price(plane,235.9375))
vero	philippe	accept	content(price(plane,235.9375))

**Fig. 7.** Conversation between Philippe and Vero as generated in the 3APL communication viewer by the piece of code given in this paper (thus fully reproducible)

and the planning rules base of 3APL agents do not support update in the present implementation, we had to encode plans and (sub)goals as beliefs (through the predicates *achieve* and *goal* respectively).<sup>6</sup> Once Vero is aware that she can achieve her goal to go to Sydney by booking a plane she issues a request for proposal for that new solution. The agents then start a new bargaining on that item and a deal is concluded. Figure 7 indicates the actual message exchange that occurs between the two 3APL agents when running them.

Note that if one changes the preferences of the agents about the price of a car rental so that they intersect, then the first bargaining dialogue would have been concluded without entering IBN, as expected with the chosen strategy.

## 6 Conclusion and Discussion

In this paper, we introduce reframing, which is the most common interest-based negotiation strategy. This paper advances the state of the art by providing a ready-to-use specification of reframing as a way to enhance a monotonic bargaining negotiation strategy in case of failure. The benefit of reframing is to eventually offer an alternative to simple monotonic bargaining when it fails. In that sense, bargaining enhanced by reframing is a dominant strategy over pure bargaining, i.e. no agent loose anything by trying it and they may gain substantial benefits.

<sup>6</sup> Another way to proceed would have been to have the rule in Vero's planning rules initial base with a guard that checks if Vero is in the state of having knowledge of that rule. That solution would have been less realistic, since it would have assumed that Vero is implicitly aware of every information she can receive.

In addressing the software engineering level of negotiating agents in a symbolic environment, we raise some limitations of the 3APL language, namely the fact that the goal base, the planing goal rules and plan reasoning rules are statically defined (one can't add some planning knowledge dynamically, e.g. as a result of communication on know-how). Ongoing and future work involves collaboration to extend 3APL in order to allow for dynamic updates of 3APL agents planning rules as well as as studying and modeling IBN strategies that are not limited to reframing.

## References

1. U. Chirico. *JIP Prolog: Java Internet Prolog*.
2. M. Dastani, M. B. van Riemsdijk, and J.-J. Meyer. *Multi-Agent Programming: Languages, Platforms and Applications*, chapter Programming multi-agent systems in 3APL, pages 10–45. Springer-Verlag, 2005.
3. M. Dastani, J. vander Ham, and F. Dignum. Communication for goal directed agents. In M.-P. Huget, editor, *Communication in Multiagent Systems - Agent Communication Languages and Conversation Policies*, LNCS, pages 239–252. Springer-Verlag, 2003.
4. M. Datani, F. de Boer, F. dignum, and J.-J. Meyer. Programming agent deliberation: An approach illustrated using the 3apl language. In *Proceedings of The Second Conference on Autonomous Agents and Multi-agent Systems (AAMAS'03)*, pages 97–104, 2003.
5. P. Faratin, C. Sierra, and N.R. Jennings. Negotiation decision functions for autonomous agents. *International Journal of robotics and Autonomous Systems*, 3-4(24):159–182, 1998.
6. S. S. Fatima, M. Wooldridge, and N. R. Jennings. Optimal negotiation strategies for agents with incomplete information. In *ATAL'01*, pages 53–68, Seattle, US, 2001.
7. R. Fisher and W. Ury. *Getting to Yes: Negotiating Agreement Without Giving In*. New York: Penguin Books, 1983.
8. N. R. Jennings, P. Faratin, A. R. Lomuscio, S. Parson, C. Sierra, and M. Wooldridge. Automated negotiation: Prospects, methods, and challenges. *Journal of Group Decision and Negotiation*, 2(10):199–215, 2001.
9. I. Rahwan. *Interest-based Negotiation in Multi-Agent Systems*. PhD thesis, Department of Information Systems, University of Melbourne, Melbourne, Australia, 2004.
10. I. Rahwan, S. Ramchurn, N. Jennings, P. McBurney, S. Parsons, and L. Sonenberg. Argumentation based negotiation. *Knowledge Engineering Review*, 18(4):343–375, 2003.
11. J. Rouchier. Reimplementation of a multi-agent model aimed at sustaining experimental economic research: the case of simulations with emerging speculation. *Journal of Artificial Societies and Social Simulation*, 6(4), 2003.
12. K. M. Sim. A market-driven model for designing negotiation agents. *Computational Intelligence*, 4(18):618–637, 2002.
13. B. Vitteau and M.P. Huget. Modularity in interaction protocols. In F. Dignum, editor, *Advances in Agent Communication, International Workshop on Agent Communication Languages (AC 2003)*, volume 2922 of *Lecture Notes in Artificial Intelligence (LNAI)*, pages 291–309. Springer-Verlag, 2004.
14. P. Winoto, G. McCalla, and J. Vassileva. Non-monotonic-offers bargaining protocol. In *Proceedings of the 3rd International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 2004)*, pages 1072–1079, 2004.

# Multiagent Reinforcement Learning for a Planetary Exploration Multirobot System

Zhang Zheng<sup>1,2</sup>, Ma Shu-gen<sup>1,2,3</sup>, Cao Bing-gang<sup>1</sup>, Zhang Li-ping<sup>1,2</sup>, and Li Bin<sup>2</sup>

<sup>1</sup> School of Mechanical Engineering, Xi'an Jiaotong University,  
Xi'an 710049, China

{Zhangzh, inte-cao}@mail.xjtu.edu.cn

<sup>2</sup> Robotics Laboratory, Shenyang Institute of Automation,  
Chinese Academy of Sciences, Shenyang 110016, China  
libin@ms.sia.ac.cn

<sup>3</sup> COE Research Institute, Ritsumeikan University,  
Kusatsu, 525-8577 Japan  
shugen@fc.ritsumei.ac.jp

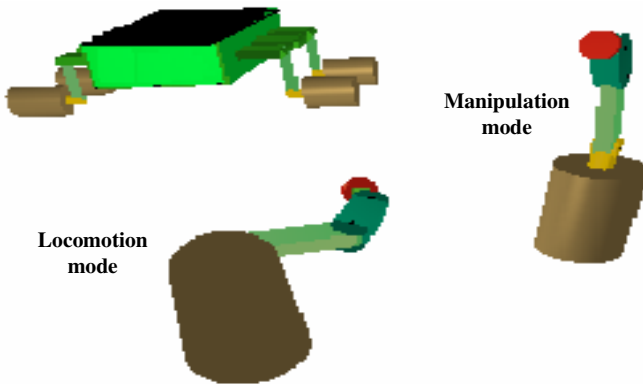
**Abstract.** In a planetary rover system called “SMC rover”, the motion coordination between robots is a key problem to be solved. Multiagent reinforcement learning methods for multirobot coordination strategy learning are investigated. A reinforcement learning based coordination mechanism is proposed for the exploration system. Four-robot climbing a slope is studied in detail as an instance. The actions of the robots are divided into two layers and realized respectively, which simplified the complexity of the climbing task. A Q-Learning based multirobot coordination strategy mechanism is proposed for the climbing mission. An OpenGL 3D simulation platform is used to verify the strategy and the learning results.

## 1 Introduction

A single robot is no longer the best solution for many of the new application domains; instead, teams of robots are required to coordinate intelligently for successful task execution. Multirobot Systems can often be used to fulfill the tasks that are difficult to be performed by an individual robot, especially in the presence of uncertainties, incomplete information, distributed control, and asynchronous computation, etc. It is impossible to predict all the potential situation robots may encounter and specify all robot behaviours optimally in advance. Therefore, robots in multirobot systems have to learn from, and adapt to their operating environment and their counterparts. Thus control and learning become two important and challenging problems in this research field [1][2].

Planetary rover systems are expected with their high abilities to undergo various missions on other planets. This paper investigates a planetary rover system called “SMC rover”, developed by Hirose, etc [3][4]. SMC consists of six wheels to support the main body, on which the main control system of SMC, solar battery cell, communication devices, tool changers for the child rovers and other equipments needed for the planetary exploring carried. Each child rover has a wheel for locomotion and an

arm for manipulation. Each of the child rovers can be looked as an autonomous child robot, which is detachable from the main body, as shown in Fig.1. The main body can be regarded as a father robot. When a child robot is in connected mode, it acts as one of the wheels of the father robot. When it is in autonomous mode, it can disconnect automatically from the main body and then rove on the planetary ground to sample or manipulate, which called locomotion mode and manipulation mode. When a child robot blocked by an obstacle in locomotion mode, it will send messages to get help from other child robot or father robot. Two or more child robots can connect automatically to form a new robot by grasping another child robot. The new one has more ability to get across the obstacle. Adopting multiple child robots, SMC rover achieves fault tolerance and high reliability.



**Fig. 1.** Model of the SMC rover system

Reinforcement learning (RL) has been an active research area in artificial intelligence especially in multiagent learning for many years. Although RL based multiagent cooperation strategy has been studied by many researchers recent years, few of them focus on the coordination strategy learning of a real multirobot system.

In this paper, we develop a coordination strategy learning mechanism for SMC rover system by using some recent results of multiagent research. A four-robot climbing slope is described in detail to explain the multirobot coordination strategy learned from RL.

## 2 Multiagent Reinforcement Learning

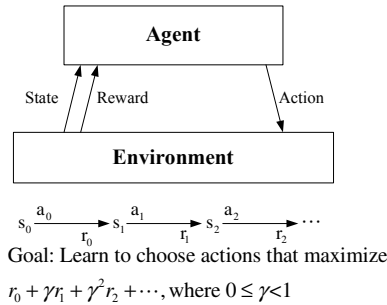
Multiagent systems and multirobot systems have both received considerable attention during the last decade. However, there have still been many challenging issues in these fields [5][6].

### 2.1 Reinforcement Learning

Reinforcement learning is learning how to map situations to actions, so as to maximize a numerical reward or cost signal. The learner is not told which actions to take,



as in most forms of machine learning, but instead must discover which actions yield the most reward by trying them. In the most interesting and challenging cases, actions may affect not only the immediate reward but also the next situation and, through that, all subsequent rewards. These two characteristics trial-and-error search and delayed reward are the two most important distinguishing features of reinforcement learning. In the standard reinforcement learning model, an agent is connected to its environment via perception and action, as depicted in Fig.2 [7] [8][9].



**Fig. 2.** An agent interacting with its environment. The agent exists in an environment described by some set of possible states  $S$ . It can perform any of a set of possible actions  $A$ . Each time it performs an action  $a_t$  in some state  $s_t$  the agent receives a real-valued reward  $r_t$  that indicates the immediate value of this state-action transition. This produces a sequence of states  $s_i$ , actions  $a_i$ , and immediate rewards  $r_i$ , as shown in the figure. The agent's task is to learn a control policy,  $\pi : s \rightarrow A$ , that maximizes the expected sum of these rewards, with future rewards discounted exponentially by their delay.

## 2.2 RL Based Multiagent Learning

In a multiagent system, they learn not only by trial and error, but also through cooperation by sharing instantaneous information, episodic experience and learned knowledge. Using independent agents as a benchmark, Ming Tan[10] studied multiagent systems in following ways, sharing sensation, sharing episodes and sharing learned policies. And the conclusions were that (a) additional sensation from another agent is beneficial if it can be used efficiently, (b) sharing learned policies or episodes among agents speeds up learning at the cost of communication; and (c) for joint tasks, agents engaging in partnership can significantly outperform independent agents although they may learn slowly in the beginning.

The difference between single-agent and multiagent system exists in the environments. In multiagent systems other adapting agents make the environment no longer stationary, violating the Markov property that traditional single agent behavior learning relies upon [2]. For individual robot learning, the traditional Q-learning has been successfully applied to many paradigms. Over the last decade many researchers have made efforts to use the RL methodology, particularly the Q-learning framework as an alternative approach to the learning of multiagent systems.

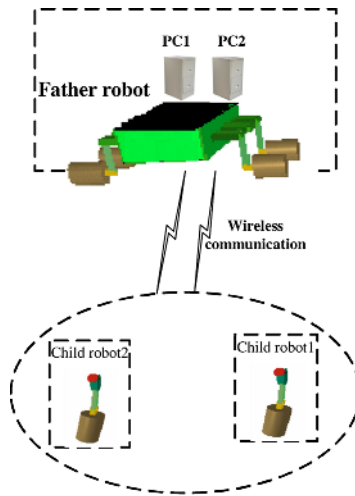
### 2.3 RL Based Multirobot Coordination

Multiagent reinforcement learning for multirobot systems is a challenging issue in both robotics and artificial intelligence. These challenges often involve the realization of basic behaviours, such as trajectory tracking, formation-keeping control, and collision avoidance, or allocating tasks, communication, coordinating actions, team reasoning, etc [2].

## 3 Multirobot Coordination in SMC

### 3.1 Control System Structure of SMC

The overall control structure of SMC is shown in Fig.3. Father robot equipped with two high performance computers PC1 and PC2. PC1 and PC2 can be a back-up system for each other. While these two computers are normal, they take charge of the tasks of themselves respectively. While one of them fail to function, the other one will immediately take over the failure computer's tasks, in case of malfunction of SMC. Therefore, the reliability of SMC is guaranteed.



**Fig. 3.** Overall control structure of SMC

The main functions of father robot include the monitoring of the whole system, task decomposition, generating of the coordination strategy, and the communication between father robot with child robots and space station. Each child robot equipped powerful Microprogrammed Control Unit (MCU) which takes charge of the controlling of each joint of the arm and adjusting the joint angle to perform a certain task. The MCU goes with some sensors and wireless communication facility to provide communication and coordination between child robots[11][12].

### 3.2 OpenGL Based Simulation Platform

Simulating platform plays a basic role in multirobot research, as a tool to verify new concepts, strategies, and algorithms quickly and efficiently. An OpenGL and Visual C++ based modeling method had been used to establish a platform for 3D simulating of multiple mobile robots and their collaboration, especially for SMC system. The cooperation strategies, the control methods and the communication mechanisms of multirobot system can be explored and verified through the platform [11].

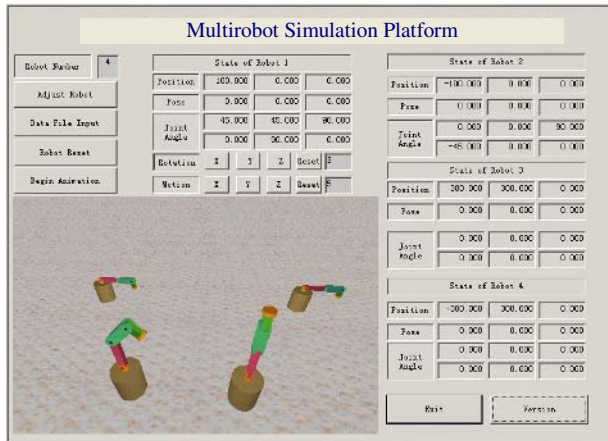


Fig. 4. Multirobot Simulation Platform of SMC

The main window of the platform is dialog based, as shown in Fig.4. It can not only show the OpenGL animation but also display the robot state numbers. The main window shows the information about position and posture of all the data dynamically. The OpenGL animation is driven by data file, which generated according to diverse methods by researcher. The OpenGL updates window each of the sampling time. Meanwhile, we can adjust the angle of view in the scene.

### 3.3 RL Based Cooperation Mechanism of SMC

According the characteristics of SMC and using RL as core algorithm, we propose a mechanism for SMC coordination, as shown in Fig.5.

After a new command transmitted to SMC from ground station, father robot will create a new mission for SMC to execute. The mission is composed of several parts, such as expected environment state, cooperative type, number of cooperative robots. These three subparts, combining with basic actions library of robot and the initial environment state act as the input of reinforcement learning block. Then the output of RL is the cooperative strategy for the multirobot to execute the specific mission. After the mission is fulfilled successfully, all the robots are waiting for a new mission transferred from father robot.

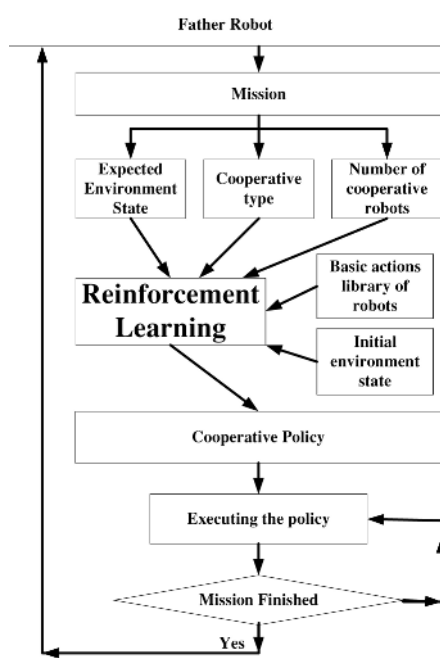


Fig. 5. RL based cooperation mechanism of SMC

## 4 Four Child Robots Climbing a Slope

### 4.1 Task Description

Fig.6 shows that when a single child robot4 cannot climb up a slope, other three robots connected with it end to end and help it to overcome this obstacle.

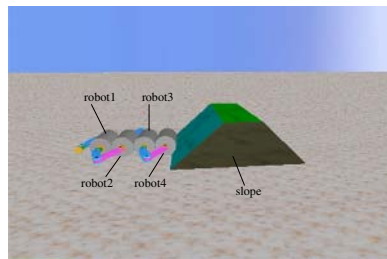


Fig. 6. Four child robot climbing a slope

### 4.2 Structure of Multirobot

In order to help robo4 climb up the slope successfully, the distance between each two child robot should be adjusted during the process of climbing, such as d32 and d43 in Fig.7.

Therefore, the inverse kinematics calculation is inescapable. The reference frame of a single robot is shown in Fig.8. The D-H parameters are shown in table.1.Where  $L1$  to  $L6$  are 60cm, 10cm, 60cm, 40cm, 20cm and 5cm respectively.

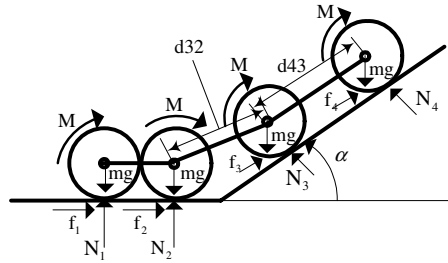


Fig. 7. Static analysis of climbing

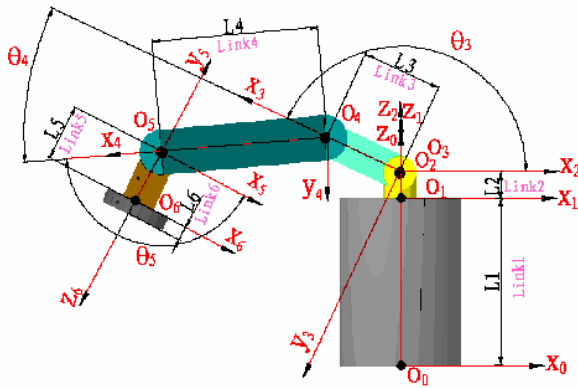


Fig. 8. Reference frame of child robot

Table 1. D-H parameters of child robot

$i$	$a_{i-1}$	$\alpha_{i-1}$	$d_i$	$\theta_i$
1	0	0	L1	$\theta_1$
2	0	0	L2	$\theta_2$
3	0	$\pi/2$	0	$\theta_3$
4	L3	0	0	$\theta_4$
5	L4	0	0	$\theta_5$
6	0	$\pi/2$	L5	$\theta_6$
H	0	0	L6	0

### 4.3 Static Analysis

By static analyzing, we found the structure that robot2 and robot1 stay under the bottom of slope can provide a maximum climbing force for robot4. We also obtained an expression,

$$\mu \geq \frac{\sin \alpha}{1 + \cos \alpha} . \tag{1}$$

Where  $\mu$  is a friction coefficient between child robot with the slope,  $\alpha$  is the maximum gradient of the slope the multirobot team can climb, as shown in Fig.9. Suppose the maximum gradient for a single robot is  $\beta$  ,

$$\beta = \arctan(\mu) . \tag{2}$$

Therefore, we can easily find that  $\alpha$  is greater or equal to  $\beta$  .

### 4.4 Q-Learning Based Coordination Strategy

Because of the complexity of the coordination task, we should make some assumptions for the task. Firstly, we suppose that each robot can perceive all the state of the environment and itself. Secondly, a robot can control and realize all the basic actions under mechanical constraints. These actions include the joint angle adjustment and dependable communication between each other [12]. Thirdly, the environment is a definite Markov discrete model. Therefore, robots select their actions only by the current state. Finally, the robot actions library is discrete. These assumptions not only avoid the huge learning space but also assure the efficiency of RL learning.

The learning method is described as following.

(1) Slope and robot parameters

The gradient of the slope is 41.8 degree. The length of the slope is 150cm. The radius of the child robot is 25cm.

(2) Simplification of the robot actions

The actions of the robots are separated into two layers. The first layer is the coordination actions of robot3 and robo4 by adjust their joints angle, the second layer is the enlarging of d32 and d43. The coordination strategy is derived by Q-learning. The second layer needs the inverse kinematics operation.

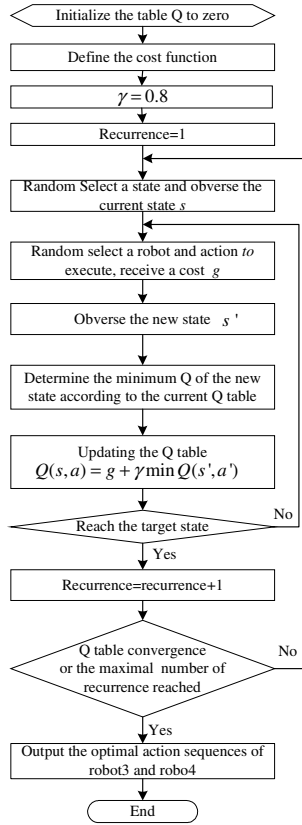
Here, we suppose that the steps of enlarging d32 and d43 are both equal to 0.2cm. Therefore, the process of climbing can be divided into 202 episodes. Only one action can be executed at one time.

(3) Cost functions

Because the roles of robot3 and robo4 are different during climbing, we gave the cost function as following,

$$g(d32) = 100 \sin\left(\frac{\pi}{2}(d32 - 50)/(98 - 50)\right) + 30 , \tag{3}$$

$$g(d43) = 60 \sin\left(\frac{\pi}{2}(d32 - 50)/(98 - 50)\right) + 20 . \tag{4}$$



**Fig. 9.** Q-learning for climbing a slope

(4) Discount factor,  $\gamma = 0.8$ .

(5) Learning algorithm

The flow chart of Q-learning for the climbing task is shown in Fig.9. Let us define the evaluation function  $Q(s, a)$ , so that its value is the maximum discounted cumulative reward that can be achieved starting from state  $s$  and applying action  $a$  as the first action. In other words, the value of  $Q$  is the reward or cost received immediately upon executing action  $a$  from state  $s$ , plus the value of following the optimal policy thereafter[9].

## 5 Results and Simulations

### 5.1 Learning Results

During the learning process, the row of Q-table is 20706, and the column of Q-table is 2. The number of the outer recurrence reaches to 0.18 millions. We encoded the action of d32 as “1”, and action of d43 as “2”.

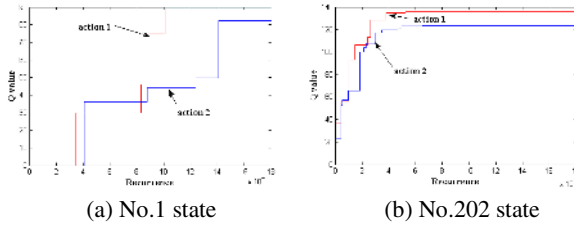


Fig. 10. The learning curves of two specified state

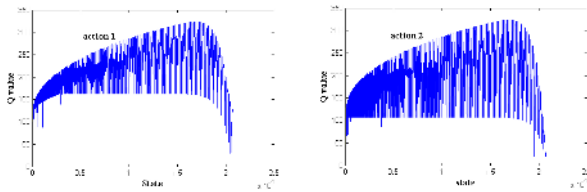


Fig. 11. The Q value of different action in each state

From the results of the learning, we can find that action 2 was selected 142 times, and that action 1 was selected 60 times. Under the optimal action sequences, all the cost reached to be 9442.1 according to the cost function. The learning curves of Q value are shown in Fig.10 and Fig.11.

### 5.2 OpenGL Based 3D Dynamic Simulation

An OpenGL based 3D simulation platform was developed to verify the results of Q-learning.

From the simulation, we found that robot4 climbed up the slope successfully under the coordination strategy learned by Q-learning, as shown in Fig.12. The positions

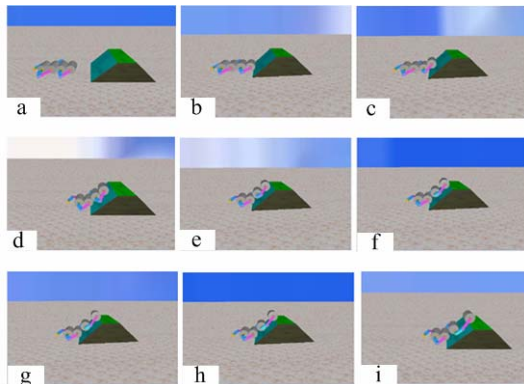
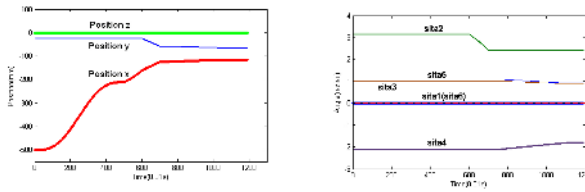
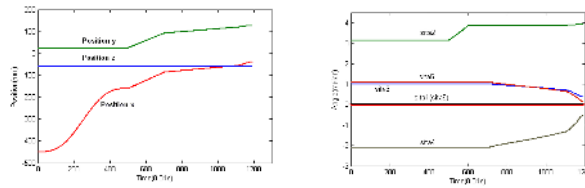


Fig. 12. OpenGL based simulation of climbing slope





**Fig. 13.** Positions and joint angles of robot3



**Fig. 14.** Positions and joint angles of robot4

and joint angles were adjusted according to the Q-learning results and inverse kinematics when climbing the slope that shown in fig.13 and fig.14.

## 6 Conclusion

In this paper, the control method and the coordination strategy of SMC system are introduced. An OpenGL based 3D simulation platform was developed to test the learning results. A Q-learning based multirobot coordination strategy was proposed and explained by four-robot climbing a slope. The learning and simulation results both proved the feasibility and practicability of the multirobot coordination mechanism.

## Acknowledgement

This work is partially supported by the National High Technology Research and Development Program of China (2002AA422130).

The authors would like to thank Prof. Shigeo Hirose, Dr Atsushi Kawakami, and Mr. Kazuhiro Matamoras in Tokyo Institute of Technology for cooperating in this research.

## References

1. M. B. Dias and A. Stentz. A market approach to multirobot coordination. Technical Report CMU-RI-TR-01-26, Robotics Institute, Carnegie Mellon University, 2001.
2. Erfu Yang and Dongbing Gu. Multiagent Reinforcement Learning for Multi-Robot Systems: A Survey. Technical Report CSM-404, Department of Computer Science, University of Essex, 2004.

3. A.Kawakami, A.Torii, K.Motomura, and Shigeo Hirose, "SMC Rover : Planetary Rover with Transformable wheels", *Experimental Robotics VIII, Advanced Robotics Series*, Bruno Siciliano Ed. Springer, pp498-506 (2003).
4. R. Damoto, A. Kawakami and S. Hirose. Study of super-mechano colony: concept and basic experimental set-up. *Advanced Robotics*, vol. 15, no. 4, pp. 391-408, April 2001.
5. Y. Shoham, R. Powers, and T. Grenager. Multi-agent reinforcement learning: a critical survey. Technical report, Stanford University, 2003.
6. Jiming Liu, XiaoLong Jin, Shiwu Zhang. *Autonomous Agents and Multi-Agent Systems: Explorations in Learning, Self-Organization and Adaptive Computation*. Tsinghua University Press. Nov. 2003.
7. Sutton, R.S., and A.G. Barto, 1998. *Reinforcement Learning: An introduction*, Cambridge, MA: MIT Press.
8. L.P. Kaelbling, M.L. Littman, and A.W. Moore. Reinforcement learning: A survey. *Journal of Artificial Intelligence Research*, 1996.
9. Tom M. Mitchell. *Machine Learning*. China Machine Press. March, 2003.
10. M. Tan. Multi-agent reinforcement learning: Independent vs. cooperative agents. In *Proceedings of the Tenth International Conference on Machine Learning*, pages 330-337, 1993.
11. Z. Zhang, S. G. Ma, B. Li, L.P. Zhang and B.G. Gang. OpenGL Based Experimental Platform for Simulation of Reconfigurable Planetary Robot System. *Journal of System Simulation*, vol.17, no.4, pp. 885-888, April 2005.
12. Z. Zhang, S. G. Ma, B. Li, L.P. Zhang and B.G. Cao. Communication Mechanism Study of a Planetary Exploration Multi-Robot System. *Robot*, vol.28, no.3, pp. 309-315, May 2006.

# An Improved Multi-agent Approach for Solving Large Traveling Salesman Problem

Yu-An Tan<sup>1</sup>, Xin-Hua Zhang<sup>2</sup>, Li-Ning Xing<sup>3</sup>,  
Xue-Lan Zhang<sup>1</sup>, and Shu-Wu Wang<sup>1</sup>

<sup>1</sup> Department of Computer Science and Engineering, Beijing Institute of Technology, Beijing 100081, China  
victortan@yeah.net, xlzh\_9@yahoo.com.cn, wsw@bit.edu.cn

<sup>2</sup> Information management college, Shandong Economic University,  
Jinan 250014, P.R. China  
zxhl2778@163.com

<sup>3</sup> The Department of Management, School of Information System and Management,  
National University of Defense Technology, Changsha 410073, P.R. China  
xln\_2002@nudt.edu.cn

**Abstract.** The traveling salesman problem (TSP) is a very hard optimization problem in the field of operations research. It has been shown to be NP-hard, and is an often-used benchmark for new optimization techniques. This paper proposes an improved multi-agent approach for solving large TSP. This proposed approach mainly includes three kinds of agents with different functions. The first kind of agent is conformation agent and its function is generating the new solution continuously. The second kind of agent is optimization agent and its function is optimizing the current solutions group. The third kind of agent is refining agent and its function is refining the best solution from the beginning of the trial. At the same time, there are many sub-agents in each kind of agent. These sub-agents accomplish the task of its superior agent cooperatively. At the end of this paper, the experimental results have shown that the proposed hybrid approach has good performance with respect to the quality of solution and the speed of computation.

## 1 Introduction

The most important member of the large set of combinatorial optimization problems is undoubtedly the traveling salesman problem (TSP), which involves the task of determining a route among a given set of nodes with the shortest possible length [1]. The study of this problem has attracted several researchers in various fields such as artificial intelligence, biology, mathematics, physics, and operations research, and there is a very large amount of literature that exists on the problem. Although it is easily formulated, it involves all aspects of combinatorial optimization and has served and continues to serve as a benchmark problem to which to apply new algorithms, including ant system (AS) [2], evolutionary methods [3], neural networks [4], tabu search (TS) [5], simulated annealing (SA) [6], genetic algorithm (GA) [7], and greedy algorithm [8].

The usual ways of solving the TSP are based either on integer linear programming techniques or on heuristic algorithms [9]. The former approach pursues the solution of the problem up to optimality. For example, highly optimized exact algorithms based on the branch-and-cut method [10] have been proposed that enable even large TSP instances to be solved. Unfortunately, this is not always possible because of the increase in computational work with problem size. Some heuristic approaches, however, have been proved to be very effective both in terms of execution times and quality of the solutions achieved. The heuristic methods do not generally yield the optimal solution but a suboptimal solution that is in most cases fairly close to the optimum. In many cases, the whole solution procedure involves two stages: firstly one heuristic constructs an initial tour that may be relatively far from the optimum, and then the initial tour is refined by means of a tour improvement heuristic.

A broad taxonomy of TSP heuristics distinguishes between local search approaches exploiting problem-domain knowledge and classical problem-independent heuristics. Domain-specific heuristics, such as 2-Opt [11], 3-Opt [12], and Lin-Kernighan (LK) [13], are surprisingly very effective for the TSP. On the other hand, general problem-independent heuristics like SA [14] and GA [15] perform quite poorly on large TSP instances. They require high execution times for solutions whose quality is often not comparable with those achieved in much less time by their domain-specific local search counterparts. Several published results demonstrate that combining a problem-independent heuristic with a local search method is a viable and effective approach for finding high-quality solutions of large TSP. The problem-independent part of the hybrid algorithm drives the exploration of the search space, thus, focusing on the global optimization task, while the local search algorithm visits the promising sub-regions of the solution space. Reference [16] proposed the chained local optimization algorithm, where a special type of 4-Opt move is used under the control of a SA schema to escape from the local optima found with LK. Reference [17] combines an original compact genetic algorithm with an efficient implementation of LK.

Among these heuristic approaches, AS is known as one of the most efficient algorithm for TSP. Inspired by the collective behavior of ant colony, DORIGO developed the AS, and later continue to design this system [18]. More recently DORIGO has been designing expanded versions of the AS paradigm. The AS is one such extension and has been applied to the symmetric and asymmetric TSP with excellent results. Reference [19] presents a new meta-heuristic approach called ACOMAC algorithm for solving the TSP, they introduce multiple ant clans' concept from parallel genetic algorithm to search solution space utilizing various islands to avoid local minima and thus can yield global minimum for solving the TSP. The main novel idea introduced by ant algorithms is the synergistic use of cooperation among many relatively simple agents which communicate by distributed memory implemented as pheromone deposited on edges of a graph [20]. Recently, the ant colony system (ACS) meta-heuristic has been proposed, representing a unifying framework to support most applications of ant algorithms to combinatorial optimization problems.

In addition to the classical heuristic algorithms of operations research, there have also been several methods based on artificial neural networks which solve the TSP [21-22]. Kohonen network incorporating explicit statistics (KNIES) has been

successfully implemented to solve both the Euclidean TSP (KNIES\_TSP) [23] and Euclidean HPP (KNIES\_HPP) [24]. Reference [25] introduces a new all-neural decomposition heuristic based on KNIES for the Euclidean TSP. Reference [26] presents an efficient multi-valued Hopfield network for the TSP; this multi-valued neural approach is superior to the best neural network currently reported for this problem. At the same time, the other technologies, such as data smoothing (also called search space smoothing and fine-tuned learning in the literature) [27], guided local search [28] and annealing-based heuristics [29] etc., have worked well when applied to the TSP.

Since the TSP has proved to belong to the class of NP-hard problem, heuristics and meta-heuristics occupy an important place in the methods so far developed to provide practical solutions for large instances [30]. Accordingly, this paper proposes an improved multi-agent approach for solving large TSP. This proposed approach mainly includes three kinds of agents with different function. The first kind of agent is conformation agent and its function is generating the new solution continuously. The second kind of agent is optimization agent and its function is optimizing the current solutions group. The third kind of agent is refining agent and its function is refining the best solution from the beginning of the trial. At same time, there are many sub-agents in each kind of agent. These sub-agents accomplish the task of its superior agent cooperatively. The rest of this paper is organized as follows. Section 2 describes the improved multi-agent approach for large TSP proposed by this paper. Section 3 presents the experimental results with 10 well-known TSP to show the performance of the proposed new hybrid approach. Finally, the conclusions about the hybrid approach are presented in Section 4.

## 2 The Proposed Multi-agent Approach

In this section, we will describe the multi-agent approach for large TSP proposed by this paper. There are seven parts in this section, (1) the architecture of this proposed multi-agent approach, (2) conformation agent, (3) optimization agent, (4) refining agent, (5) ant colony system agent, (6) genetic algorithm agent and (7) fast local searching agent.

### 2.1 The Architecture of This Proposed Multi-agent Approach

Figure 1 depicts the framework of the architecture of this proposed multi-agent approach. From figure 1, we can see that this proposed approach mainly includes three kinds of agents with different function. The first kind of agent is conformation agent and its function is generating the new solution continuously. The second kind of agent is optimization agent and its function is optimizing the current solutions group. The third kind of agent is refining agent and its function is refining the best solution from the beginning of the trial. The proposed approach is terminated when one of the following criteria (end condition of whole approach) is satisfied: (1) maximum preset search time is exhausted; (2) the known optimal solution was achieved by this proposed approach.

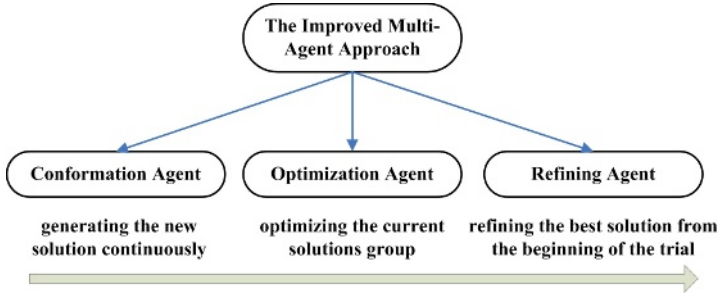


Fig. 1. The framework of this proposed multi-agent approach

### 2.2 Conformation Agent

The first kind agent of this proposed multi-agent approach is conformation agent and its function is generating the new solution continuously. Figure 2 depicts the architecture of the conformation agent. In this paper, there are two different sub-agents, ant colony system (ACS) agent and genetic algorithm (GA) agent, in the architecture of the conformation agent. These two sub-agents accomplish the task of conformation agent cooperatively. About the ACS agent and GA agent, we will describe them in the following parts. As you know, we can insert other excellent agent in this conformation agent architecture; it improves the expansibility of this proposed multi-agent approach.

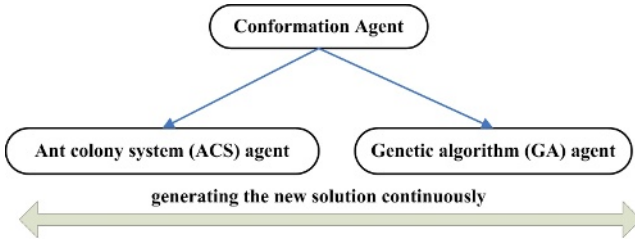


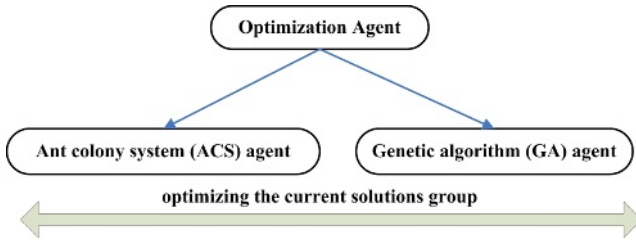
Fig. 2. The architecture of the conformation agent

### 2.3 Optimization Agent

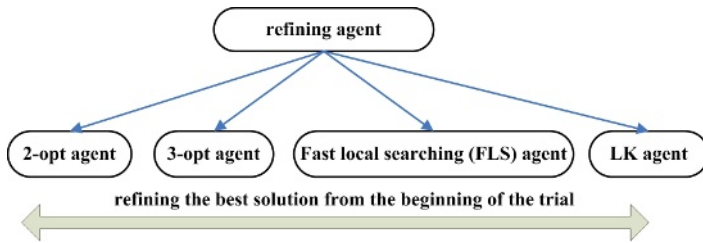
The second kind agent of this proposed multi-agent approach is optimization agent and its function is optimizing the current solutions group. Figure 3 depicts the architecture of the optimization agent. In this paper, there are two different sub-agents, ACS agent and GA agent, in the architecture of the optimization agent. These two sub-agents accomplish the task of optimization agent cooperatively. Similarly, the future new agent can be integrated into this optimization agent architecture too.

### 2.4 Refining Agent

The third kind agent of this proposed multi-agent approach is refining agent and its function is refining the best solution from the beginning of the trial. Figure 4 depicts the



**Fig. 3.** The architecture of the optimization agent



**Fig. 4.** The architecture of the refining agent

architecture of the refining agent. In this paper, there are four different sub-agents, 2-opt agent, 3-opt agent, fast local searching (FLS) agent and LK agent, in the architecture of the refining agent. Limit to length, we don't describe the detail of these four sub-agent. As a representative example, here we will describe the detail of the FLS agent in the following. Similarly, the future new agent can be integrated into this optimization agent architecture too.

## 2.5 Ant Colony System (ACS) Agent

Ant Colony System (ACS) differs from the previous ant system because of three main aspects: (1) the state transition rule provides a direct way to balance between exploration of new edges and exploitation of a priori and accumulated knowledge about the problem, (2) the global updating rule is applied only to edges which belong to the best ant tour, and (3) while ants construct a solution a local pheromone updating rule (local updating rule, for short) is applied. As was the case in ant system, ants are guided, in building their tours, by both heuristic information (they prefer to choose short edges), and by pheromone information: An edge with a high amount of pheromone is a very desirable choice. The pheromone updating rules are designed so that they tend to give more pheromone to edges which should be visited by ants. The implement process of the ACS agent is showed as figure 5.

## 2.6 Genetic Algorithm (GA) Agent

GA is one of the evolutionary search methods that can provide optimal or near optimal solutions for the combinatorial optimization problems. The most attractive features of

<u>The Ant Colony System Agent</u>
<b>INPUT:</b> (1) the city coordinates of the TSP, (2) interrelated parameter.
<b>OUTPUT:</b> a group of new solution.
<b>IMPLEMENT PROCESS:</b>
Initialize
Each ant is positioned on a starting node
<b>Loop</b> /*at this level each loop is called a step */
Each ant builds a solution by applying a state transition rule
A local pheromone updating rule is executed
<b>Until</b> all ants have built a complete solution
The global pheromone updating rule is applied
The Pheromone decay rule is executed
Save the solution group gained by the ant colony system agent

**Fig. 5.** The implement process of the ant colony system agent

<u>Genetic Algorithm (GA) Agent</u>
<b>Initialization;</b>
$t \leftarrow 0$ ;
Set population size $pop\_size$ , number of generation $max\_gen$ , probability of crossover $p_c$ , and probability of mutation $p_m$ ;
Initialize parent population $P(t)$ ;
<b>Evaluate</b> $P(t)$ and select the best solution $\sigma^*$ with the minimum objective function among $P(t)$ ;
<b>WHILE</b> (no termination criteria) <b>DO</b>
<b>Regenerate</b> $C(t)$ from $P(t)$ by applying the crossover and mutation operations;
<b>Evaluate</b> $C(t)$ and select the current best solution $\sigma$ with the minimum objective function among $C(t)$ ;
<b>Update</b> the best solution $\sigma^*$ , i.e., if $\sigma < \sigma^*$ , then $\sigma^* = \sigma$ ;
<b>Select</b> $P(t+1)$ from $P(t)$ and $C(t)$ ;
$t \leftarrow t+1$ ;
<b>END WHILE.</b>

**Fig. 6.** The overall procedure of the genetic algorithm agent

GA are the flexibility of handling various kinds of objective functions with fewer requirements on fine mathematical properties. It has been applied to a number of fields like engineering, biology, computer science, and social sciences. The main issues in developing a GA-based algorithm are chromosome representation, initialization of the population, evaluation measure, crossover, mutation, and selection strategy. Also, the genetic parameters such as population size  $pop\_size$ , number of generation  $max\_gen$ , probability of crossover  $p_c$ , and probability of mutation  $p_m$ , should be determined before execution of a GA. Let  $P(t)$  and  $C(t)$  be, respectively, populations for parent and offspring in generation  $t$ . Overall procedure of the used genetic algorithm agent is described in Fig. 6.

### 2.7 Fast Local Searching (FLS) Agent

The fast local searching agent works as follows. The current solution is broken down into a number of small sub-neighborhoods and an activation bit is attached to each one of them. The idea is to scan continuously the sub-neighborhoods in a given order, searching only those with the activation bit set to 1. These sub-neighborhoods are



called active sub-neighborhoods. Sub-neighborhoods with the bit set to 0 are called inactive sub-neighborhoods and they are not being searched. The neighborhood search process does not restart whenever we find a better solution but it continues with the next sub-neighborhood in the given order. Initially, all sub-neighborhoods are active. If a sub-neighborhood is examined and does not contain any improving moves then it becomes inactive. Otherwise, it remains active and the improving move found is performed. Depending on the move performed, a number of other sub-neighborhoods are also activated. In particular, we activate all the sub-neighborhoods where we expect other improving moves to occur as a result of the move just performed. As the solution improves the process dies out with fewer and fewer sub-neighborhoods being active until all the sub-neighborhood bits turn to 0. The solution formed up to that point is returned as an approximate local minimum. The implement process of the fast local searching agent is showed as figure 7.

<b>The Fast Local Searching Agent</b>
<b>INPUT:</b> the best solution from the beginning of the trial.
<b>OUTPUT:</b> a new best solution from the beginning of the trial.
<b>IMPLEMENT PROCESS:</b>
<b>Loop</b> /*at this level each loop is called an iteration */
Select an active sub-neighborhood randomly
Generate the set of moves in the giving sub-neighborhood and join them into a null population
<b>Loop</b> /*at this level each loop is called a step */
Each move in the temporary population is selected to operate
Execute this certain move to the initial solution and get a new solution
Select a better solution from the initial solution and the new one to replace the initial solution
<b>Until</b> all the move in the temporary population have operated
If the initial solution was not improved, then set the giving sub-neighborhoods inactive
<b>Until</b> all the sub-neighborhoods are inactive
Save the new best solution fast local searching agent

Fig. 7. The implement process of the fast local searching agent

### 3 Performances Evaluation

The proposed approach was tested on 10 TSP benchmark problems in which the numbers of cities ranged from 1'000 to 15'000. All of these test problems are selected from TSPLIB [21]. The method was implemented on a Pentium IV 2.4 GHz personal computer with a single processor and 512M RAM. Each problem was run 20 times. Table 2 presents the experimental results obtained by applying the proposed approach to these 10 problems. This proposed approach found the optimal tours of all tested problems except problem D15112. The optimal solution to the small problems (in which the number of cities is less than 3'000) is determined in each trial. For large problems, such as PCB3038 and FNL4461, this proposed approach found the optima at least 16 times out of 20 independent trials. The best solution found by this proposed approach to problem D15112 is 1'576'117, which is only 0.0025% above the optimum (1'573'084). For each test problem, the average tour length does not exceed 0.0549% above the optima.

**Table 1.** Experimental result obtained by this proposed approach to 10 TSP problems

Sequence Number	Problem Name	Avg. Tour Length (Error %)	Optimization Time (Sec)	Optimal Times
1	PR1002	259045(0)	120.77	20
2	PCB1173	56892(0)	111.55	20
3	U1432	152970(0)	139.83	20
4	VM1748	336556(0)	187.88	20
5	U2152	64253(0)	278.87	20
6	PR2392	378032(0)	273.64	20
7	PCB3038	138561.4(0.0063)	805.18	19
8	FNL4461	182575.5(0.0052)	3152.6	16
9	USA13509	19987674.9(0.0241)	45814	11
10	D15112	1573947.6(0.0549)	53407	0

**Table 2.** The 6 different methods used in this section

Mark	Name of the optimization algorithm	Reference
M1	Chained local optimization algorithm	Reference [16]
M2	Compact genetic algorithm with LK	Reference [17]
M3	ACOMAC algorithm	Reference [18]
M4	Guided local search with FLS-2Opt	Reference [28]
M5	Annealing-based heuristics	Reference [29]
M6	The improved multi-agent approach	This paper

**Table 3.** The experimental results obtained form 10 test problems using the 6 methods

TSP	Performance Index	M1	M2	M3	M4	M5	M6
PR1002	Average time (S)	152.16	142.35	137.52	132.95	135.85	120.77
(259045)	Average error (%)	0.00440	0.00429	0.00410	0.00392	0.00370	0
PCB1173	Average time (S)	140.54	131.48	127.02	122.8	125.48	111.55
(56892)	Average error (%)	0.00702	0.00684	0.00654	0.00625	0.00590	0
U1432	Average time (S)	176.17	164.82	159.22	153.93	157.29	139.83
(152970)	Average error (%)	0.00868	0.00846	0.00809	0.00773	0.00730	0
VM1748	Average time (S)	236.71	221.45	213.94	206.83	211.34	187.88
(336556)	Average error (%)	0.00726	0.00707	0.00676	0.00646	0.00610	0
U2152	Average time (S)	351.34	328.7	317.55	307	313.69	278.87
(64253)	Average error (%)	0.01083	0.01055	0.01009	0.00964	0.00910	0
PR2392	Average time (S)	344.75	322.54	311.59	301.24	307.81	273.64
(378032)	Average error (%)	0.02177	0.02121	0.02029	0.01938	0.01830	0
PCB3038	Average time (S)	1014.4	949.06	916.85	886.4	905.72	805.18
(137694)	Average error (%)	0.03200	0.03117	0.02983	0.02848	0.02690	0.0063
FNL4461	Average time (S)	3971.9	3716	3589.8	3470.6	3546.3	3152.6
(182566)	Average error (%)	0.05782	0.05632	0.05389	0.05146	0.04860	0.0052
USA13509	Average time (S)	57720	54001	52168	50435	51535	45814
(19982859)	Average error (%)	0.09851	0.09595	0.09181	0.08767	0.08280	0.0241
D15112	Average time (S)	67286	62951	60814	58794	60076	53407
(1573084)	Average error (%)	0.21760	0.21194	0.20280	0.19365	0.18290	0.0549

This proposed approach was compared to five methods (see table 2), including the chained local optimization algorithm [16], the compact genetic algorithm with LK [17], the ACOMAC algorithm [18], guided local search with FLS-2Opt [28], and the annealing-based heuristics [29]. These methods performed well on these test problems, according to a survey of the relevant literature.

The experimental results obtained for 10 test problems, using these six different methods, are given in table 3. Here, the average optimization time and the average optimization error were used for evaluating these different methods. Average optimization time is the average time of the computation time of these 20 independent runs. In the same way, average optimization error is the average error of the computation error of these 20 independent runs. The computation error is the relative error between the optimal result achieved by giving method and the optimal result produced by the recent published heuristic optimization algorithm. From table 3, we can see that both the average optimization time and the average optimization error, the proposed method of this paper are better than the other four methods. Simulations have shown that the proposed hybrid approach for the TSP has excellent performance with respect to the quality of solutions and the speed of calculation.

## 4 Conclusions

The contribution of this paper is summarized as following. (1) This paper presents a new multi-agent architecture for the TSP; the future new agent can be integrated into this architecture. This proposed approach mainly includes three kinds of agents with different functions. The first kind of agent is conformation agent and its function is generating the new solution continuously. The second kind of agent is optimization agent and its function is optimizing the current solutions group. The third kind of agent is refining agent and its function is refining the best solution from the beginning of the trial. This proposed approach supports the distributed solving to the TSP. (2) This study designs many sub-agents in each kind of agent, these sub-agents accomplish the task of its superior agent cooperatively. At the end of this paper, the experimental results have shown that the proposed hybrid approach has good performance with respect to the quality of solution and the speed of computation.

## References

1. DEINEKO V.G., HOFFMANN M., OKAMOTO Y., et al.: The Traveling Salesman Problem with Few Inner Points. *Operations Research Letters*. 34(1) (2006) 106-110
2. CARO G.D., DORIGO M.: AntNet: Distributed Stigmergetic Control for Communications Networks. *Journal of Artificial Intelligence Research*. 9(12) (1998) 317-365
3. TSAI H.K., YANG J.M., TSAI Y.F., et al. An Evolutionary Algorithm for Large Traveling Salesman Problems. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*. 34(4) (2004) 1718-1729
4. LEUNG K.S., JIN H.D., XU Z.B.: An Expanding Self-organizing Neural Network for the Traveling Salesman Problem. *Neurocomputing*. 62(1) (2004) 267-292

5. GENDREAU M., LAPORTE G., SEMET F.: Tabu Search Heuristic for the Undirected Selective Traveling Salesman Problem. *European Journal of Operational Research*. 106(2) (1998) 539-545
6. BUDINICH M.: Self-organizing Neural Network for the Traveling Salesman Problem that is Competitive with Simulated Annealing. *Neural Computation*. 8(2) (1996) 416
7. WANG Y.P., HAN L.X., and LI Y.H.: A New Encoding based Genetic Algorithm for the Traveling Salesman Problem. *Engineering Optimization*. 38(1) (2006) 1-13
8. AHUJA R.K., ORLIN J.B., TIWARI A.: Greedy Genetic Algorithm for the Quadratic Assignment Problem. *Computers and Operations Research*. 27(10) (2002) 917-934
9. LAPORTE G.: The Traveling Salesman Problem: an Overview of Exact and Approximate Algorithms. *European Journal of Operational Research*. 59(2) (1992) 231-247
10. PADBERG M., RINALDI G.: Optimization of a 532-city Symmetric Genetic Traveling Salesman Problem by Branch and Cut. *Operational Research Letters*. 6(1) (1987) 1-7
11. CROES G.A.: A Method for Solving Traveling Salesman Problems. *Operational Research*. 6(6) (1958) 791-812
12. LIN S.: Computer Solution of the Traveling Salesman Problem. *Bell System Technology Journal*. 44(2) (1965) 2245-2269
13. LIN S., KERNIGHAN B.W.: An Effective Heuristic Algorithm for the Traveling Salesman Problem. *Operational Research*. 21(2) (1973) 498-516
14. MARTIN O., OTTO S.W.: Combining Simulated Annealing with Local Search Heuristic. *Annual Operational Research*. 63(2) (1996) 57-75
15. MICHALEWICZ, Z.: *Genetic Algorithms + Data Structures = Evolution Programs*. 3rd edn. Springer-Verlag, Berlin Heidelberg New York (1996)
16. MARTIN O., OTTO S.W., FELTEN E.W.: Large Step Markov Chain for the Traveling Salesman. *Journal of Complex System*. 5(3) (1991) 299
17. BARAGLIA R., HIDALGO J.I., PEREGO R.: A Hybrid Heuristic for the Traveling Salesman Problem. *IEEE Transactions on Evolutionary Computation*. 5(6) (2001) 613-622
18. DORIGO M., MANIEZZO V., COLORNI A.: The Ant System: Optimization by a Colony of Cooperating Agent. *IEEE Transactions on System, Man, and Cybernetics, Part B: Cybernetics*. 26 (1) (1996) 29-42
19. TSAI C.F., TSAI C.W., TSENG C.C.: A New Hybrid Heuristic Approach for Solving Large Traveling Salesman Problem. *Information Sciences*. 166(1) (2004) 67-81
20. DORIGO M., GAMBARDELLA L.M.: Ant Colony System: A Cooperative Learning Approach to the Traveling Salesman Problem. *IEEE Transactions on Evolutionary Computation*. 1(1) (1997) 53-66
21. REINELT G.: TSPLIB - A Traveling Salesman Problem Library. *ORSA Journal on Computing*. 3(4) (1991) 376-384
22. SMITH K.: Neural Networks for Combinatorial Optimization: A Review of More Than a Decade of Research. *INFORMS Journal Computer*. 11(1) (1999) 15-34
23. ARAS N., OOMMEN J., ALTINEL I.K.: Kohonen Network Incorporating Explicit Statistics and Its Application to the Traveling Salesman Problem. *Neural Networks*. 12(9) (1999) 1273-1284
24. ALTINEL I.K., ARAS N., OOMMEN J.: Fast, Efficient and Accurate Solutions to the Hamiltonian Path Problem Using Neural Approaches. *Computer and Operations Research*. 27(5) (2000) 461-494
25. ARAS N., ALTINEL I.K., OOMMEN J.: A Kohonen-Like Decomposition Method for the Euclidean Traveling Salesman Problem-KNIES\_DECOMPOSE. *IEEE Transactions on Neural Networks*. 14(4) (2003) 869-890

26. CASERMEIRO E.M., MARIN G.G., PEREZ J.M.: An Efficient Multi-valued Hopfield Network for the Traveling Salesman Problem. *Neural Processing Letters*. 14(2) (2001) 203-216
27. COY S.P., GOLDEN B.L., WASIL E.A.: A Computational Study of Smoothing Heuristics for the Traveling Salesman Problem. *European Journal of Operational Research*, 124(2) (1999) 15-27.
28. VOUDOURIS C., TSANG E.: Guided Local Search and Its Application to the Traveling Salesman Problem. *European Journal of Operational Research*, 113(1) (1999) 469-499
29. PEPPER J.W., GOLDEN B.L., WASIL E.A.: Solving the Traveling Salesman Problem with Annealing-Based Heuristics: A Computational Study. *IEEE Transactions on Systems, Man, and Cybernetics Part A: Systems and Humans*. 32(1) (2002) 72-77
30. BURIOL L., FRANCA P.M.: A New Memetic Algorithm for the Asymmetric Traveling Salesman Problem. *Journal of Heuristics*. 10(5) (2004) 483-506

# Design of Agent Registry/Repository System Based on ebXML

Il Kwang Kim<sup>1</sup>, Jae Young Lee<sup>1</sup>, and Il Kon Kim<sup>2</sup>

<sup>1</sup> Intelligent System Lab, Department of Computer Science, Kyungpook National University, Daegu, South Korea

{dinosa, intvis}@daum.net

<sup>2</sup> Intelligent Health Information Sharing System Development Center, School of Medicine, Kyungpook National University, Daegu, South Korea

ikkim@mail.knu.ac.kr

**Abstract.** The goal of this paper is to propose a way to register agents so that makes users or autonomous agents to find target agents more conveniently and efficiently. In order to achieve this goal, we have survey the key technologies such as UDDI, ebXML, FIPA DF. As the results, first we have designed a new agent registry/repository system based on ebXML technologies. Next, we introduce our agent registry/repository system for wide area network and describe how to use our system for registering agents and searching target agents for communications. Finally, we give an example to illustrate a typical scenario in which user receives the results from our proposed agent registry/repository system.

## 1 Introduction

The web based Internet environment provides that it can use diverse online services through Network. Furthermore, the computing environment has been quickly changing into focusing on the Web, and many technologies that related to the network have been studied. The distributed computing environment based on network, and technology has been advanced to CORBA, JAVA RMI, and Web service based on SOAP, and even to the grid computing. One of the main fields that many researches have been studied and that contains paradigm reflecting the trend of distributed environment is the agent area. Especially, the study of agent, agent platform, and mobility of agent and the availability of agent have established through many discussion. The Fig. 1 shows roadmap of technologies that related to the agent infrastructure construction [1]. These technologies can be used by various purposes for construction of agent infrastructure. Already, IBM Aglet, Ajanta, Voyager, GrassHopper, Tracy, and many more agent platforms have been studied and developed based on these technologies. And in addition, the study of agent based middleware for context-aware service have been issued recently [2, 3, 4, 5].

These trends announces era of agents that all work can be treated by agent in the near future and we can use services that are offered by agent. Related to these trend, this paper propose a new agent registry/repository system based on ebXML, in which the system can identify, register, and store the agent created in various platforms.

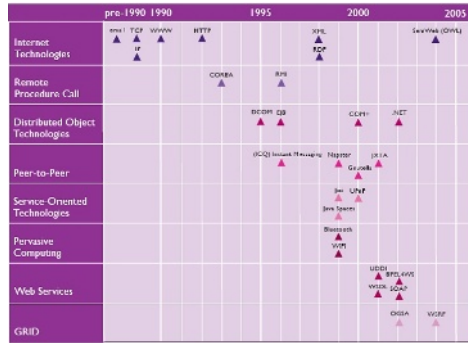


Fig. 1. Agent-related technologies for infra support

The proposed system can be compared to directory service, it can identify, register and store the agent created in multi-platform environment (different OS, Machines, Agent Systems, etc) and can also support for searching target agents for communication. The Fig. 2 shows the final goal of this study. There exist a number of distributed registries and repositories and the code of the agent that is created in the agent platform is to be stored into repository and metadata is to be registered into registry. This structure supports the transmission of agent code and the federation of distributed repositories through registry query.

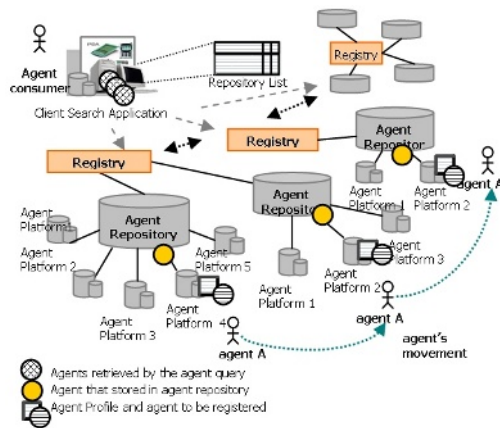


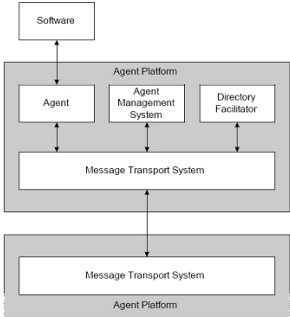
Fig. 2. Distributed Agent Registry / Repository Service

## 2 Related Works

### 2.1 FIFA Specification

In this section, we will briefly describe the characteristics of FIPA, multi-agent platform standard, and the background of the architecture of the agent registry / repository system based on ebXML that is similar to FIPA DF (Directory Facilitator). FIPA

has worked on standardization of FIPA97, FIFPA98, FIPA2002 and related standards as categorized by Applications, Abstract Architecture [6], Agent Message Communication, Agent Management, and Agent Message Transport. The Fig. 3 shows the reference model of FIPA Agent Management [7]. As this diagram explains itself, agent platform is composed of AMS (Agent Management System), DF (Directory Facilitator), and MTS (Message Transport System).



**Fig. 3.** FIPA Agent Management Reference Model

AMS deals with agent management, DF acts as yellow page. MTS controls the message transport among internal agents of platform as well as external agents existed even in different platform. In this case, MTS checks the AID (Agent ID) and route the message to the target agent. The registration of the agent occurs through AMS and the only the agent registered using AMS can use service of agent platform. Once it was registered, it can send and receive the message to and from the different agent or can advertised itself using DF.

**2.2 ebXML vs. UDDI**

The proposed system has similar characteristics to FIPA DS as register and advertises the agent. However, in addition to those capabilities, the proposed system can act as agent code server when applied to mobile agent platform.

In order to achieve this goal, the registry / repository system is necessary for created agent to transport, store, and manager in secure manner. Especially, in order to classify and utilize agents properly, it is also necessary to support effective classification scheme. In line with these purports, the ebXML’s RIM was used in this study [8, 9, 10, 11]. The ebXML is being under the process of protocol revision centering on international standardization institutes such as UN/CEFACT and OASIS, after its v1.0 was produced in May 2005. Actually, it has been recognized as the international standard in B2B filed. At first, the ebXML was begun with intend to make a global and unified e-market place. The XML-based B2B standards have been produced meanwhile, e.g., xCBL, RosettaNet, eCo, cXML, etc., but it was difficult to share inter-local standard’s information in as much as those had respective protocol. The ebXML overcome suchlike local standards’ limitation and presented all standards related to B2B, such as electronic transaction, protocol, communication, contents,



business process, etc. Out of such ebXML standards, the RIM has standard data system suitable to the repository and the RS (Registry Service) defining the standard interface such as register, inquiry, modification, deletion, etc. in the repository. The RS has OMS (Object Management Service) and OQMS (Object Query Management Service); OMS is to manage the life cycle of objects. Although RIM v3.0's draft was produced as of February 10, 2005, yet RIM v2.0 was adopted in this study following the time point of study. "RegistryObject" and "RegistryEntry" are the most important objects in RIM. The "RegistryEntry" is created at each contents submitted to the registry. ExternalLink is being to express URI (Uniform Resource Identifier) and the access path of the contents stored in outside of registry, and it can assign the related association. "Classification" and "ClassificationNode" are used to define the system that classifies the objects effectively and hierachically. AuditableEvent" is to store the audit record of objects. "User" and "Organization" are to store user's and institute's information. At first, web service's UDDI (Universal Description, Discovery and Integration)[12, 13, 14, 15] was also considered in this study, but the ebXML's RIM was adopted in the end. In case of UDDI architecture, UDDI registry is composed of general items (white page), industrial items (yellow page) and technical items (green page) and provides directory service similar to a telephone directory. However, it does not contain management part able to store (repository) and manage the more information on hospitals or enterprises. For this reason, the ebXML's RIM was adopted rather than the UDDI registry. Particularly, ebXML aims at securing the concrete protocol necessary for business transaction and at defining such protocols. Accordingly, it was judged that the ebXML is more advantageous compared with the web service, as in case of business transaction, safe trade, time control, fault tolerance, etc. are needed to interchange information. However, considering interoperability, the inter-coordination of registries that refer different information model such as ebXML and UDDI is essential. For this, ebXML V2.2 propose various methods for inter-operation among diverse ebXML registries and SUN has announced a specification of JAXR [16].

### 3 System Overview

We assume that agent mentioned in this research is active entity or object that is developed for various purposes and is independently executable on various agent platforms. Users or developers can develop agent that could perform specific task that user endowed or offer general services on specific agent platform. Developer could be the provider of agent that is registered at authorized server and he could offer this agent to users who need this service. In other words, when one is developing system, one can use agent that was developed by others for completion of the system. Of course, in this case payment method policy and downloading of agent might be necessary. Also, version control of agent is one of very important tasks.

In real e-Business environment, much effort is invested in finding of business information. Current e-Business registries are providing keyword-based searching that might have limits on searching of agents. For example, for a given condition, "Find an agent that provides credit card payment service which provides reservation of air ticket from Seoul to LA," current e-Business registry structures will find the agent

that provides reservation service first and then check other business information with detailed condition. In fact, this method makes users spent much time and effort in agent searching. So, for more effective agent searching environment, agent that intelligently search many registries and find out the target agent on behalf of humans is necessary.

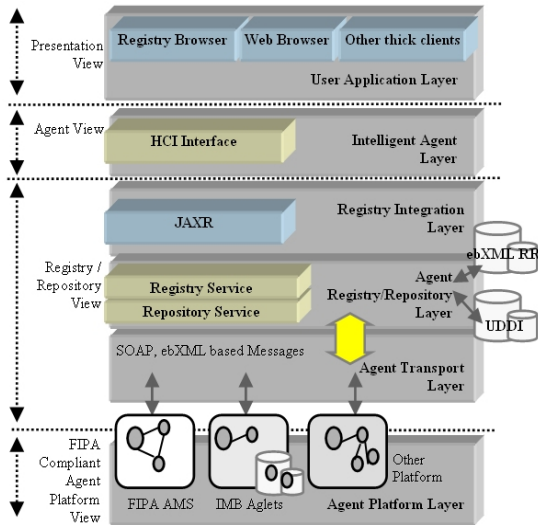
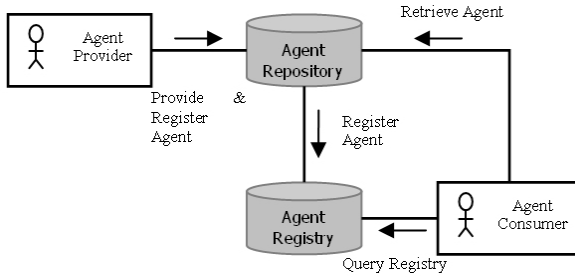


Fig. 4. Conceptual System Structured Layer

Fig. 4 presents the conceptual structured layer for registration and searching of agent derived from results that had researched so far. This structured layer has approximately four point-of-views and several layers included. Agent Platform Layer is FIPA Compatible Agent Platform and includes agent systems that each developed by different vendors. Three layers included in Registry Repository System View take on the role of providing agent registration/storing service. Agent Transport Layer takes on the role of sending agent and exchanging messages between agent platform and registry using XML, SOAP, and ebXML based message protocols. Agent Registry / Repository Service takes on the most important roles, and each has responsibility as follows.

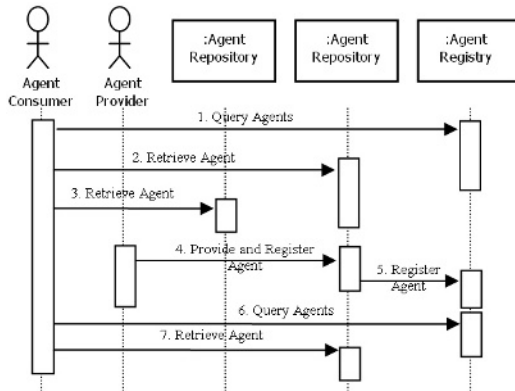
- Basically, Agent Registry Service takes on the role of keeping/managing meta-data that is needed on execution of business logic using agent, such as agent owner, company, registered agent, agent system information, etc. Agent consumer can query through Registry Service, and doesn't have to know where the agent is located.
- Repository Service takes on the role of storing agent that is provided on semi-permanent memory device securely and registering relevant information on the registry. In addition, it sends reply to agent retrieval request. This works as a kind of code server and has advantage of increasing agent transmission rate in mobile agent environment.



**Fig. 5.** Agent registration & retrieval diagram

The Fig. 5 shows transactions among Registry, Repository, and Actor. Agent Provider actor means agent creator/provider and is user or specific Agent System that has responsibility of sending Agent to Agent Repository actor. Agent Consumer actor can be user or other agent that asks Agent Registry actor about relevant question. Agent Consumer actor can attempt to directly connect to agent on other agent system through results from Agent Registry actor, or if necessary, through more than one repository target agent can be provided.

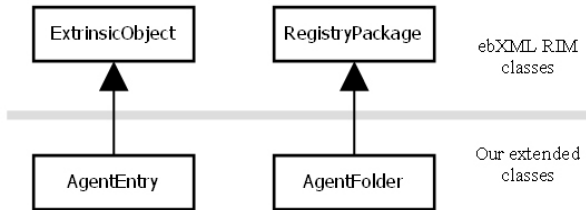
Fig. 6 describes the process using Sequence Diagram of UML notation more specifically. Registry Integration Layer provides searchable base that integrate heterogeneous structured registry (such as UDDI Registry, ebXML Registry, or Registry that other vendor provides, etc.) information. Intelligent Agent Layer analyzes ontology on integrated registry or makes index of knowledge with relevant information for categorizing and using of registered agents. Also, using HCI (Human Computer Interaction) interface, this layer connects to user application layer and accepting various requests from users, and then returns processed result that is acquired from querying Registry Information Layer to User Application Layer. User Application Layer could be user application such as thin client (web browser) or thick client (custom Registry browser).



**Fig. 6.** Agent registration & retrieval sequence diagram

## 4 Key Design

In this section, we describe the key contents of designed system based on content that have introduced so far. Fig. 7 explains an extended class diagram in the system based on RIM of ebXML.



**Fig. 7.** Extended classes from ebXML RIM

As we know through the figure, each AgentEntry and AgentFolder is extended from ExtrinsicObject and RegistryPackage respectively. AgentEntry is composed of normal and common properties to register and search agent and a language for AID (Agent ID), Agent Registration Time, Agent Communication. And AgentFolder is also composed of the property, that is, the unique ID of folder, ID of owner, the last update time to save and store the registered Agent in specific Repository as the safe method. [Table 1] shows the principal attributes of AgentEntry. AID is type of ID that is composed of the name of agent and the address of agent platform connected by '@', and Agent Unique ID is unique ID for purpose of management by registry. Agent Execution Side Type indicates whether the agent works in the Client application (at Client-sided) or Server-sided application. Agent Collaboration Type expresses whether the registered agent works as stand-alone or needs to be collaborated to others.

**Table 1.** Attributes of AgentEntry

Attributes
AID (Agent Id)
Agent Registration Time
Agent Execution Side Type (Client side or Server side)
Agent Collaboration Type
Agent Home System Id
Agent Home System Name
Agent Communication Language
Agent Unique Id

Fig. 8 shows the relationship between Agent(Entry) and registry objects. Each associative relation is expressed as follows:

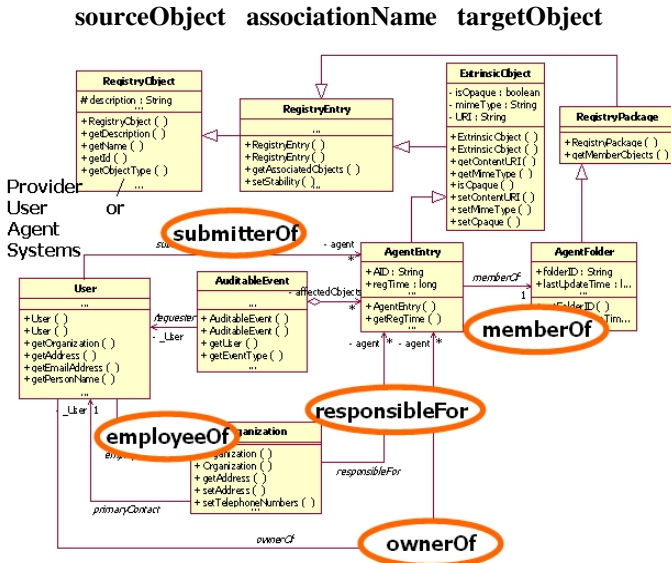


Fig. 8. Agent association with registry objects

We defined 5 basic associationName between sourceObject and targetObject. If Agent provider is A, organization belonging to user is B, Agent is C, associationName defines the following relationship and saves corresponding in formations within Registry.

A *submitterOf* C. In this case, A means a provider who transmits agent C.

A *employeeOf* B. It represents that User A who registers Agent belongs to certain organization B.

B *responsibleFor* C. It expresses that administration responsibility of registered Agent C is under organization B.

A *ownerOf* C. It supposes that registered Agent C is owned by provider A and saves the relationship in property of ownerOf.

C *memberOf* D. It represents that registered Agent C is packaged into agent repository physically and saves relationship as memberOf.

Of course, each relationship between them is designed to enable multiple registrations according to policy. These five relationships is to be stored with the information of sourceObject and targetObject in Association object of ebRIM and audit information according to this transaction is recorded in AuditableEvent object. Consequently, this enables system to search corresponding object fundamentally using only simple relationship operation between associationNames or sourceObject and targetObject.

## 5 Implementation and Results

Fig. 9 shows the system architecture for implementation based on the system described through section 3 and 4.

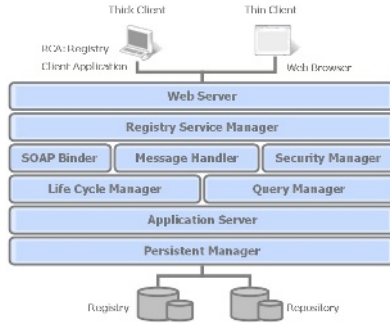


Fig. 9. Agent registry / repository software system architecture

Registry Service Manager takes charge of connection with other configuration components such as Life Cycle Manager, Query Manager, etc., and processes the request on classifying and registering Agent. SOAP (Simple Object Access Protocol) Binder and Message Handler are being used to manage SOAP message, and Security Manager takes charge of the preservation of PKI (Public Key Infrastructure)-based XML documents.

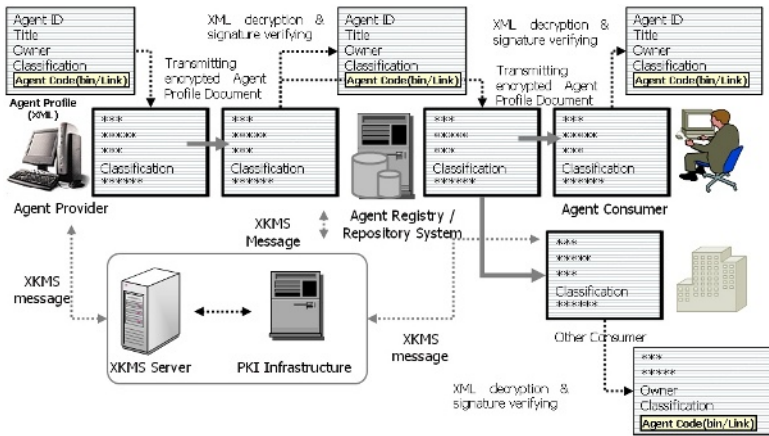


Fig. 10. Overview of agent profile document encryption and decryption

The Fig. 10 shows the outline of encryption or decryption for the preservation of Agent Profile, and it is being continuously studied. In case of Agent Profile, general original or voluntary data are encrypted using XML Encryption [17] technology, and Agent Profile signed using XML Digital Signature [18] technology is transmitted to Repository. In such a case, Security Manager of the system takes charge of decrypting the encrypted Agent Profile extracting information or requesting the verification of certificates through XKMS (XML Key Management System) [19] server. Additionally, Persistent Manager is to provide persistent and safe data service. LifeCycle

Manager takes charge of the life cycle of Agent Profile (Submitted-Approved-Deprecated-Removed) and QueryManager supports so that clients can rapidly retrieve Agent. In consequence of implementing the system based on the foregoing, the final screen is as follows.

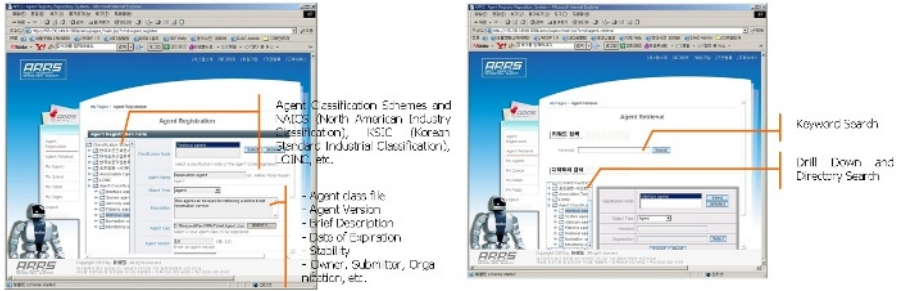


Fig. 11. Agent registering and retrieving

Fig. 11 shows the screen of registering and retrieving web-based Agent, respectively. In case a user just pushes the “transmission” button after selecting agent classification system and inputting additional information, the Agent is transmitted to the Repository server and its information is stored in the registry. Registered Agent can be searched using simple keyword search or Drill-Down search utilizing Classification Scheme.

Fig. 12 shows the screen that inquires the list of the Agent registered by such a method, and the result that was retrieved by the RCA (Registry Client Application) developed as a registry browser.

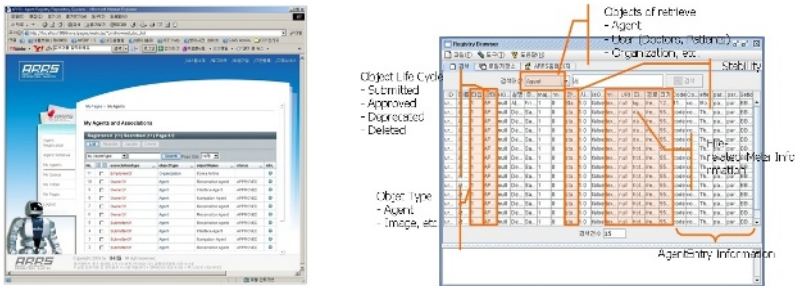


Fig. 12. Web-based agent list up and query result by our registry browser

So far, we have derived the proper relationship among Agents, suppliers of Agent, and consumer of Agent and supported the registration and management of transmitted Agent in accordance with RIM based on ebXML and classification. Through this made agent registration and search could be done without difficulty in global network environment.

## 6 Conclusion and Future Works

Up until now, we have shown the architecture and implementation of the ebXML based Agent Registry / Repository system and have proposed the one of the methods that could support the service of Agent registration, storing and search using that system.

Here we summarized the issue related to the implementation, future work and research for the proposed system.

- Autonomous Agent Registration and Retrieval: Methodology of autonomous Agent registration and retrieval system without human input
- Ontology based Agent Brokering: Ontology based Agent brokering or matchmaking system and intelligent service by the system
- Registry Federation: Inter-Operation model among Registry based on heterogeneous information model
- Agent Transport as a Code Server: Minimize the overhead of transportation of Agent by utilizing Code Server for transporting and dynamic storing of Agent Class in Mobile Agent environment
- Agent based SOA environment: Implementing Service Oriented Architecture environment based on Agent

At present, the prototype of proposed Agent Registry / Repository System is implemented and it is focused on Agent Registration and search by user. However, with this fundamental study of our proposed system, we are planning to analyze the problem that might occur when we apply this system to real situation in order to solve the issues described above. And we are also planning to measure the result that we could obtain when we inter-operate our system with legend various agent platforms. In addition, our next research will focus on designing and developing the optimized Agent Query Model for supporting ad-hoc query service using Filter Query. Also in order to support mobile agent environment, performance test of Agent Code Server in real situation is necessary and the method of using agent tracking service needs to be researched and developed.

**Acknowledgement.** This study was supported by a grant of the Korea Health 21 R&D Project(A05-0909-A80405-05N1-00000A), Ministry of Health & Welfare, Republic of Korea.

## References

1. Technology Roadmap: Overview and Consultation Report, *Agent based computing*. AgentLink, (2004)
2. Danny B. Lange and Mitsuru Oshima.: Programming and Deploying Java™ Mobile Agents with Aglets. Addison-Wesley Professional, (1998)
3. Tracy Technical Report No 7, A Decisive Agent Based Exchange Platform for Tracy Mobile Agent Systems. January (2005)
4. Arkady Zaslavsky.: Mobile Agents-Can They Assit with Context Awareness?. Proceeding of IEEE international conference on mobile Data management, (2004)



5. Paolo Bellavista, Dario Bottazzi, Antonio Corradi, Rebecca Montanari and Silvia Vecchi.: Mobile Agent Middleware for Context-aware Applications. Handbook of Mobile Computing, (2004)
6. *FIPA Abstract Architecture Specification*, FIPA TC Agent Management, <http://www.fipa.org>. (2002).
7. *FIPA Agent Management Specification*, FIPA TC Architecture, <http://www.fipa.org>. (2002).
8. ebXML Specification, Available at: <http://www.ebxml.org/specs/index.htm>. (2005).
9. ebXML RIM V2.0 <http://www.oasis-open.org/committees/regrep/documents/2.0/specs/ebrim.pdf>
10. 10 ebXML RS V2.0 <http://www.oasis-open.org/committees/regrep/documents/2.0/specs/ebrs.pdf>
11. OASIS Standards and Other Approved Work. <http://www.oasis-open.org>
12. UDDI Specification. [http://www.oasis-open.org/committees/tc\\_home.php?wg\\_abbrev=uddi-spec](http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=uddi-spec)
13. Web Services Architecture, W3C Working Group Note. <http://www.w3.org/TR/ws-arch>
14. W3C Web Services Activity. <http://www.w3.org/2002/ws>
15. Service Oriented Architecture and Web Service, IBM. <http://www-306.ibm.com/software/solutions/webservices/documentation.html>
16. SUN, Java API for XML Registries, <http://java.sun.com/webservices/jaxr/index.jsp>
17. XML Encryption Syntax and Processing W3C Recommendation 10 December 2002. <http://www.w3.org/TR/xmlenc-core>
18. XML-Signature Syntax and Processing W3C Recommendation 12 February 2002. <http://www.w3.org/TR/xmldsig-core>
19. XML Key Management Specification (XKMS) Version 2.0, W3C Editor's Draft 30th March 2005. <http://www.w3.org/2001/XKMS/>

# Ant Agent-Based QoS Multicast Routing in Networks with Imprecise State Information

Xin Yan and Layuan Li

Department of Computer Science, Wuhan University of Technology, Wuhan, P.R. China  
yanxin@mail.whut.edu.cn

**Abstract.** The existing schemes based on ant agents don't take into account the impact of the imprecision of network state information on routing performance. In this paper, we design a novel ant agent-based multicast routing algorithm with bandwidth and delay guarantees, called QMRA, which works for packet-switching networks where the state information is imprecise. In our scheme, an ant uses the probability that a link satisfies QoS requirements and the cost of a path instead of the ant's trip time or age to determine the amount of pheromone to deposit, so that it has a simpler migration process, less control parameters and can tolerate the imprecision of state information. Extensive simulations show our algorithm can achieve low routing blocking ratio, low average packet delay and fast convergence when the network state information is imprecise.

## 1 Introduction

With the rapid development of communication networks (including circuit-switching networks and packet-switching networks), more intelligent techniques are needed to deal with the complexity problems of network routing such as QoS-aware routing, network dynamics and load balancing, and improve the reliability and scalability of network routing. The network routing problem is intrinsically a distributed, dynamic and multi-objective problem; as well as the routing decisions should only be made on the basis of the local state information by each network node. These features make it well adapted to solve the routing problems mentioned above by using ant agent approaches [1].

Ant agents, also called mobile agents based on swarm intelligence or ant colony optimization, can be used to make a distributed and adaptive network routing model, which is inspired by the ability of discovering the shortest path to a food source and the trail-laying/trail-following behaviors of a real ant colony. A reinforcement learning technique is applied to the routing model so that ant agents can collaboratively find the optimal path between a source node and a destination node through the positive feedback mechanism [2].

At present, there exist several schemes to apply ant agents to network routing. In [3], an ant-based control (ABC) system was designed to solve the load-balancing problem in circuit-switching networks by Schoonderwoerd et al. It can efficiently cope with the dynamic aspects of network routing problem, but doesn't work in asymmetric networks

any more. Caro and Dorigo's AntNet was originally designed for the network routing in packet-switching networks [4]. Unlike the ABC system, AntNet isn't only restricted to the routing applications in symmetric networks. Nonetheless, the routing in AntNet is determined by means of a very complex procedure and involved in too many control parameters. In [5], an agent-based routing system (ARS) with bandwidth and hop count constrains was proposed by K. Oida and M. Sekido as an extended version of AntNet. As such, on the basis of AntNet, G.Y. Lu and Z.M. Liu presented a multicast routing algorithm with delay and delay variation constraints [6].

Being similar to traditional networks, the state information in the networks based on ant agents is inherently imprecise due to the non-negligible link propagation delay and the dynamic variation of traffic load [7]. The imprecision has a noticeable impact on not only the speed of discovering the optimal path that satisfies certain QoS constrains but also the routing success ratio, because ants maybe probe the links that can't really meet the QoS requirements. However, the ant agent-based schemes mentioned above do NOT take into account the imprecision of state information while constructing their network routing models.

In this paper, we will discuss the QoS-aware multicast routing based on ant agents, with a consideration of the influence of state information imprecision on its routing performance. The rest of the paper is organized as follows. Section 2 describes a system model. Section 3 proposes our scheme. Section 4 gives some analysis and proofs about our algorithm. Some experimental results are provided in Section 5. Section 6 sums up our work.

## 2 System Model

A communication network is usually represented as a weighted, connected graph  $G=(V,E)$ , where  $V$  denotes the set of nodes and  $E$  denotes the set of full-duplex, directed communication links connecting the nodes.  $|V|$  and  $|E|$  denote the number of nodes and links in the network, respectively. Without loss of generality, only simple graphs are considered, in which there exists at most one link between each pair of ordered nodes.

### 2.1 Pheromone Table Structure

In order to apply ant agents to packet-switching network routing, we replace the traditional routing tables in network nodes by tables of probabilities, called "pheromone tables", as the pheromone strengths are represented by these probabilities. Suppose that a node  $i$  ( $i \in V$ ) with  $k$  neighbor nodes in a packet-switching network possesses a pheromone table  $R_i = [r_{d,j}^i]_{|V|-1,k}$  with  $|V|-1$  rows and  $k$  columns. Each row in the pheromone table corresponds to a destination node and each column corresponds to a neighbor node. The value  $r_{d,j}^i$  expresses the probability of choosing neighbor  $j$  as the next hop on the way to destination  $d$ . In a pheromone table, all of the probability values in each row must conform to the constraint:

$$\sum_{j \in J(i)} r_{d,j}^i = 1 \quad (1)$$

where  $J(i)$  denotes the set of node  $i$ 's neighbors, and  $d \in V$ . An example of pheromone tables is shown in Fig. 1.

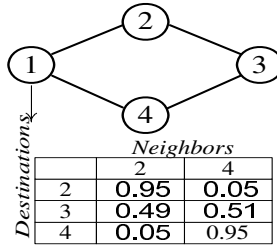


Fig. 1. An example of pheromone tables

### 2.2 Routing Model

Suppose that  $s \in V$  is the source node of a multicast tree, and  $M \subseteq \{V - \{s\}\}$  is a set of destination nodes of the multicast tree. Let  $R^+$  be the set of positive real numbers. For any link  $e \in E$ , we define the link state information: the bandwidth function  $bw(e): E \rightarrow R^+$ , the delay function  $delay(e): E \rightarrow R^+$ , and the cost function  $cost(e): E \rightarrow R^+$ . Similarly, for any node  $n \in V$ , we define the node state information: the delay function  $delay(n): V \rightarrow R^+$ , and the cost function  $cost(n): V \rightarrow R^+$ . We use  $T(s, M)$  to denote a multicast tree that has the following relations:

$$\begin{aligned}
 bw(p(s, d)) &= \min\{bw(e), e \in p(s, d)\} \\
 delay(p(s, d)) &= \sum_{e \in p(s, d)} delay(e) + \sum_{e \in p(s, d)} delay(n) \\
 cost(T(s, M)) &= \sum_{e \in T(s, M)} cost(e) + \sum_{e \in T(s, M)} cost(n)
 \end{aligned}
 \tag{2}$$

where  $p(s, d)$  denotes the path from the source  $s$  to a destination  $d$  of  $T(s, M)$ . Let  $B$  and  $D$  be the minimal residual bandwidth and the maximal end-to-end delay requirements to  $p(s, d)$ , respectively. The problem of the multicast routing with bandwidth and delay constraints can be represented as follows.

$$\begin{aligned}
 bw(p(s, d)) \geq B \wedge delay(p(s, d)) \leq D \\
 \wedge cost(T(s, M)) = \min[...].
 \end{aligned}
 \tag{3}$$

### 2.3 Imprecise State Information

As mentioned before, the state information in the networks based on ant agents is dynamic and imprecise. It includes: a) the connectivity of network topology; b) link state information, which consists of the residual bandwidth on a link, the propagation delay along the link, and the link cost; c) node state information, which consists of the queuing delay at a node and the node cost. For the purpose of simplicity, we only take

into account the imprecision of the residual bandwidth on a link. Such a simplification will not degrade the routing performance significantly, in that the residual bandwidth is the most representative among the above state information.

In order to capture the imprecision of link residual bandwidth,  $\forall(i, j) \in E$ , we can use  $\Delta b(i, j)$  to denote the estimated maximum fluctuation of link residual bandwidth  $b(i, j)$ . That is, based on the recent state history, the actual residual bandwidth of link  $(i, j)$  is expected to be between  $b(i, j) - \Delta b(i, j)$  and  $b(i, j) + \Delta b(i, j)$  while data packets passing. We'll try to calculate  $\Delta b(i, j)$  through the following way. Let  $\Delta b_{old}(i, j)$  and  $\Delta b_{new}(i, j)$  be the values of  $\Delta b(i, j)$  when a forward ant and its backward ant arrive respectively (see the next session for details). Similarly, let  $b_{old}(i, j)$  and  $b_{new}(i, j)$  be the values of  $b(i, j)$  when a forward ant and its backward ant arrive respectively. The value of  $\Delta b_{new}(i, j)$  is calculated as follows.

$$\Delta b_{new}(i, j) = \alpha \times \Delta b_{old}(i, j) + (1 - \alpha) \times |b_{new}(i, j) - b_{old}(i, j)|, \tag{4}$$

where factor  $\alpha (\alpha < 1)$  is a control parameter that determines how fast history information  $\Delta b_{old}(i, j)$  is forgotten. It's usually set for 0.25 in our simulation experiments.

We further assume that the actual residual bandwidth on link  $(i, j)$  at the time of routing is uniformly distributed in the range of  $[b_{new}(i, j) - \Delta b_{new}(i, j), b_{new}(i, j) + \Delta b_{new}(i, j)]$ , and that,  $B$  is the bandwidth requirement to the link. Therefore, we can calculate the probability that the link satisfies the bandwidth requirement. If  $B > b_{new}(i, j) + \Delta b_{new}(i, j)$ , then  $P(b(i, j) \geq B) = 0$ , otherwise,

$$P(b(i, j) \geq B) = \frac{b_{new}(i, j) + \Delta b_{new}(i, j) - B}{2\Delta b_{new}(i, j)}. \tag{5}$$

### 3 Proposed Algorithm

Although the multicast routing problem with two or more additive metric constraints is a NP-complete problem, we can efficiently solve it by applying a scheme based on ant agents, which is called QMRA (QoS-aware Multicast Routing based on Ant agents). Moreover, this kind of scheme has many performances that are better than the traditional heuristic methods on network dynamics, load balancing and reliability.

#### 3.1 Essential Ideas

At first, we devise two kinds of ant agents: forward ants and backward ants to adaptively discover the minimal-cost path that satisfies the bandwidth and delay requirements. The format of each ant packet is defined as Fig. 2, where each item between a pair of parentheses is the comment to the corresponding field name.

Forward ants are periodically launched from certain source to a randomly selected destination at an equal time interval  $\Delta t$ . Suppose that a forward ant travels from a source

$s$  to a destination  $d$  along path  $(s, \dots, i, j, \dots, d)$ , which is illustrated in Fig. 3. When the forward ant arrives at node  $i$ , it records the visited nodes and the cost of path  $(s, \dots, i)$ , and calculates the accumulated trip time along the path. If  $ant.time > ant.dr$ , the forward ant dies, otherwise it starts the following operations.

$ant.ID$ (ant agent's identifier)
$ant.node$ (visited node set)
$ant.cost$ (cost set of visited paths)
$ant.time$ (trip time record)
$ant.bw$ (bandwidth set of visited links)
$ant.br$ (bandwidth requirement)
$ant.dr$ (delay requirement)
$ant.s$ (source node)
$ant.d$ (destination node)

Fig. 2. The format of each ant packet

Assume that node  $j$  will be considered as the next node to the destination by the forward ant, if it has the maximum probability value in the pheromone table maintained at node  $i$  and  $j \notin ant.node$ . The forward ant begins to detect the current residual bandwidth of link  $(i, j)$  as  $b_{old}(i, j)$ , collects its old residual bandwidth value that was detected by the previous backward ant and cached at node  $i$ , and calculates the value of  $\Delta b_{old}(i, j)$ . If  $b_{old}(i, j) \geq ant.br$ , node  $j$  is chosen, and  $b_{old}(i, j)$  as well as  $\Delta b_{old}(i, j)$  are put into the stack of field  $ant.bw$ , or another node that has the maximum probability value except for node  $j$  and hasn't previously been visited will be taken into account.

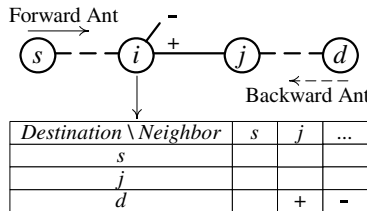


Fig. 3. The operation process of ant agents

When destination  $d$  is reached, the forward ant generates a backward ant, dumps all of its memory to the backward ant and dies. Then, the backward ant moves from destination  $d$  to source  $s$  along the same path as that of its corresponding forward ant, but in the reverse direction. Arriving at node  $i$  from node  $j$ , the backward ant detects the current residual bandwidth of link  $(i, j)$  as  $b_{new}(i, j)$ , and evaluates the values of  $\Delta b_{new}(i, j)$

and  $P(b(i, j) \geq ant.br)$  according to (4) and (5) respectively. The backward caches  $b_{new}(i,j)$  at node  $i$ , and updates the entry corresponding to destination  $d$  in the pheromone table at node  $i$  in terms of the following equations:

$$r_{d,j}^i = \frac{r_{d,j}^i + \delta r}{1 + \delta r}, r_{d,k}^i = \frac{r_{d,k}^i}{1 + \delta r} \tag{6}$$

In the equations,  $k \in J(i)$  and  $k \neq j, \delta r$  is the reinforcement parameter of pheromone.

$$\delta r = \frac{a}{c(s, j)} + b \cdot P(b(i, j) \geq ant.br) \tag{7}$$

where  $a$  and  $b$  are the system parameters,  $c(s,j)$  is the cost of path  $(s, \dots, i, j)$  stacked in the backward ant. That is, the probability of node  $i$ 's neighbor  $j$  is increased while the probabilities of the other neighbors are decreased. Furthermore, the increase is in proportion to the reciprocal of  $c(s,j)$  and the value of  $P(b(i, j) \geq ant.br)$ , which makes the later forwards ants prefer to choose the cheaper, wider and faster paths. Similarly, the backward ant carries out the above scheme at each intermediate node on its trip.

### 3.2 Detailed Description

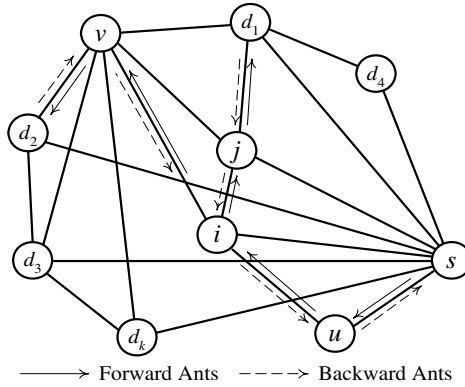
When a node receives a call request to open a multicast session with the destination set  $M$ , the bandwidth requirement  $B$  and the delay requirement  $D$ , it becomes the source  $s$  of the multicast session. The pheromone table of each node in the network is initialized using an initial value, which is determined by the current status of bandwidth distribution.

1) We select a destination  $d_1$  at random in  $M$ , and set the forward ant number for  $L$ . Before a forward ant being launched from source  $s$ , some of fields in it are initialized:  $ant.node=\{s\}$ ,  $ant.cost=\{0\}$ ,  $ant.time=0$ ,  $ant.br=B$ ,  $ant.dr=D$ ,  $ant.s=s$ ,  $ant.d=d_1$ . The forward ant starts from the source to its a neighbor node that not only satisfies the bandwidth constraint but also has the maximum probability value, as the next hop to the destination. If there're more than one neighbor node with the maximum probability value, we select one among them at random. After the next node being determined, field  $ant.bw$  in the forward ant is initialized in light of what we depicted above.

2) A forward ant moves from one node to another, one by one, as shown in Fig. 4 in which the network topology is generated by Waxman's algorithm [8]. The forward ant implements our scheme at each intermediate node during the movement towards destination  $d_1$ . As mentioned above, when the forward ant arrives at node  $d_1$ , the backward ant is generated and travels towards source  $s$ , updating the entry corresponding to node  $d_1$  in the pheromone table at each intermediate node along the path in terms of (4), (5), (6) and (7).

3) After a time interval  $\Delta t (\Delta t \ll D)$ , the second forward ant is launched from source  $s$ . And then, step 1 and 2 are repeated until the convergence (see the next section for explanations) appears in our algorithm. Eventually, the minimal-cost path  $p(s,d_1)$  satisfying the bandwidth and delay requirements is established, and all the nodes along

the path are designated as the set  $M_1$ . It should be noticed that if all of the  $L$  forward ants with the same destination  $d_1$  have been launched completely before the convergence appears, the call request will be rejected.



**Fig. 4.** Network illustration

4) Similar to step 1, we randomly select a destination  $d_2$  in set  $M - \{d_1\}$ , as such set the forward ant number for  $L$ . The fields in the forward ants are initialized:  $ant.node = \{s, M_1\}$  (though  $s \in M_1$ , it doesn't affect the operation process),  $ant.cost = \{0\}$ ,  $ant.time = 0$ ,  $ant.br = B$ ,  $ant.dr = D$ ,  $ant.s = s$ ,  $ant.d = d_2$ . Suppose that the first forward ant considers choosing the node  $u$  as its next hop to destination  $d_2$  by checking the residual bandwidth and comparing the probability values in the pheromone table. If  $u \in ant.node$ , the forward ant moves to node  $u$ , and the similar process continues until the first node  $v$  that  $v \notin ant.node$  comes forth as shown in Fig. 4. When  $v \notin ant.node$ , the forward ant travels to node  $v$  from node  $i$ , and reinitializes its fields as follows:  $ant.node = \{i, M_1\}$ ,  $ant.cost = \{0\}$ ,  $ant.time = 0$ ,  $ant.br = B$ ,  $ant.dr = D$ ,  $ant.s = i$ ,  $ant.d = d_2$ . It's noticed that forward ant at node  $i$  knows whether its next node  $v \in ant.node$  or not. From now on, the process of discovering the feasible and optimal path between node  $i$  and destination  $d_2$  corresponds with the above steps, i.e. step 1, step 2 and step 3.

5) If  $M - \{d_1, d_2, \dots, d_k\} = \Phi$ , the multicast tree is constructed, otherwise step 4 is repeated.

6) After the multicast tree is established, the source sends a PATH message to all the destinations along the route of the multicast tree to request resources for the multicast session. The RESV messages from the destinations will try to reserve resources along the reverse route of the multicast tree. The source transmits data to a certain destination immediately if the RESV message from the destination arrives at the source.

Seen from the above process, our proposed algorithm, QMRA is also an algorithm based on on-demand routing, which can decrease the needed ant number in the case of ensuring the convergence of the algorithm.



## 4 Discussion

### 4.1 Convergence Rule and Ant Number

At present, most algorithms based on ant colony optimization are prone to choose the restriction to iteration times as their convergence rule (i.e. stopping condition) [9], and make the needed ant number equal the number of nodes in networks [10].

In our scheme, the convergence rule is to see whether the source continuously receives some backward ants that travel the same path, which lasts a period of time  $D$  (i.e. the delay constraint), since it launched the first forward ant.

It's well known that it is very difficult to determine the needed ant number because more ant number can accelerate the convergence of algorithms, but will increase the overhead of networks. According to the results of our extensive simulation experiments, we give the following empirical formula to determine the ant number needed to discover a single optimal path.

$$L = \left\lceil |V| + \frac{\mu B_{max}}{B} + \frac{\eta D_{max} d_{net}}{D} \right\rceil \tag{8}$$

where each one of  $L$ ,  $|V|$ ,  $B$ , and  $D$  has the same meaning as before, and  $B_{max}$ ,  $D_{max}$ ,  $d_{net}$  denote the maximal bandwidth and delay of all links in a network topology, and the topology's diameter respectively. The control parameters in (8),  $\mu$  and  $\eta$  are usually set for 0.45 and 0.8 respectively in our simulation experiments.

### 4.2 Correctness and Complexity

In [11], T. Stützle and M. Dorigo have proved some convergence properties for a class of ant colony optimization algorithms; however, so far we haven't found out any theoretical proof on the convergence properties of the routing algorithms based on ant agents. The correctness and complexity of QMRA can be testified and analyzed by the following non-formalized theorems.

**Theorem 1.** In a network  $G=(V,E)$  applying QMRA, suppose that: a) there exists a unique minimal-cost path  $p(s,d)$  that satisfies the bandwidth and delay constraints, with  $H$  hops from the source  $s$  to the destination  $d$ ; b)  $P(t)$  is the probability that QMRA finds  $p(s,d)$  at the time  $t$ . Thus, it holds that:

$$\lim_{t \rightarrow x} P(t) = 1 \tag{9}$$

where  $x$  is a finite positive real number, and the complexity of the convergence time is  $O(|V|^2)$ .

**Proof.** Assume that  $T_{max}$  is the maximum propagation delay of all the links in the network  $G$ . Consider  $T_{fmax}$  and  $T_{bmax}$  to denote the maximum time which all the nodes in  $G$  spend processing a forward ant and its backward ant respectively. We give the proof of the theorem in manner of a recursion.

a) When the hop count equals 1, it can be concluded that the time which QMRA needs to discover  $p(s,d)$  is not more than  $L(2T_{max} + T_{f_{max}} + T_{b_{max}})$ , where  $L$  is the number of forward ants launched from the source. That is, when  $t \rightarrow x = L(2T_{max} + T_{f_{max}} + T_{b_{max}})$ ,  $\lim_{t \rightarrow x} P(t) = 1$ .

b) Further suppose that the theorem holds when the hop count is  $H-1$  ( $H > 2$ ). Considering the independence property of ants' behavior probing each link, we can conclude that the time which QMRA needs to discover  $p(s,d)$  is not over  $|E|L(2T_{max} + T_{f_{max}} + T_{b_{max}})$ , where  $|E|$  denotes the number of links in  $G$ .

c) From the above, it can be drawn that the time needed to discover  $p(s,d)$  is not more than  $(|E|+1) \times L(2T_{max} + T_{f_{max}} + T_{b_{max}})$  when the hop count of  $p(s,d)$  is  $H$ . In the preceding expression,  $T_{max}$ ,  $T_{f_{max}}$  and  $T_{b_{max}}$  can be considered as constants, and  $L$  is the linear function of  $|V|$  (see (8) for details). Hence, the time complexity that QMRA finds a single optimal path is  $O(|E||V|)$ , and the time complexity of its establishing a multicast tree is  $O(|E|^2|V|)$ . For most networks,  $|E|=O|V|$ , then  $O(|E||V|) = O(|V|^2)$ , and  $O(|E|^2|V|) = O(|V|^3)$ . The theorem holds.

**Theorem 2.** The multicast tree found by QMRA is the minimal-cost tree that satisfies the bandwidth and relay requirements.

**Proof.** As mentioned before, a forward ant will neither select the nodes visited nor the links that can't meet the bandwidth requirement, and will die if its life exceeds the delay requirement. In addition, from Theorem 1, we can see that each path found by QMRA is the minimal-cost path among the paths from the source to each destination. Therefore, the multicast tree constructed by the minimal-cost paths is the minimal-cost tree. The theorem holds.

## 5 Simulation

We implement our proposed algorithm by modifying and developing the multicast routing model in NS2, and evaluate its routing performance via two measures: routing blocking ratio and average packet delay.

$$R_{blk} = \frac{N_{rej}}{N_{req}}, D_{avg} = \frac{D_{pkt}}{N_{pkt}}, \tag{10}$$

where  $R_{blk}$  and  $D_{avg}$  denote routing blocking ratio and average packet delay respectively,  $N_{rej}$  and  $N_{req}$  are the rejected request number and the sum of call requests respectively, and  $D_{pkt}$  as well as  $N_{pkt}$  are the total delay and the total number respectively of the data packets arriving at all nodes at a time. For the sake of comparison, another ant agent-based routing system for QoS guarantees, ARS is likewise simulated. Each one of the reported values is averaged over 10 trials.

### 5.1 Experimental Environment

The network topology with 50 nodes used in our experiments is generated by Waxman's algorithm, in which the bandwidth and the delay of each link are uniformly

distributed in [10Mbps, 50Mbps] and [0.5ms, 2.5ms] respectively. For the purpose of simplicity, the cost of each link is configured to one unit.

The source node of each multicast tree is selected at random over the network. Each source node generates multicast session requests according to a Poisson process with an arrival rate  $\lambda(\lambda=1/15s)$ , with uniform random selection of destination nodes. In order to simulate the real situation, each multicast group size (i.e. the destination node number) is always less than 20% of the total node number. The total number of data packets per multicast session, their sizes and their inter-arrival times, are negative exponential distributed, with mean values of 50, 512 bytes and 28ms respectively. Ant number  $L$  is calculated in terms of (8), and the size of each ant is fixed to 256 bytes. It takes each node in the network 1ms to process each ant. System parameters,  $a$ , and  $b$  are set for 0.08 and 0.01 respectively. The bandwidth requirement  $B$  and the delay requirement  $D$  of each session request are uniformly distributed in [5Mbps, 30Mbps] and [8ms, 25ms] respectively.

In order to simulate the imprecision of network state information, we import another performance metric: imprecision rate  $\delta$  defined as follows.

$$\delta = \supremum \left\{ \frac{|b_{act}(i, j) - b_{cal}(i, j)|}{b_{cal}(i, j)} \right\} \tag{11}$$

where  $b_{act}(i, j)$  (i.e.  $b(i, j)$  in (5)) is the actual residual bandwidth value of link  $(i, j)$  while data packets passing, and  $b_{cal}(i, j)$  (i.e.  $b_{new}(i, j)$  in (5)) is its last residual bandwidth value used to calculate a route. In addition,  $b_{cal}(i, j)$  is uniformly distributed in  $[0.9C(i, j), C(i, j)]$ , where  $C(i, j)$  is the capacity of link  $(i, j)$ ; while  $b_{act}(i, j)$  is uniformly distributed in the range of  $[(1 - \delta)b_{cal}(i, j), (1 + \delta)b_{cal}(i, j)]$ .

Other parameters needed to simulate ARS are deployed as depicted in [5].

### 5.2 Experimental Results

Fig. 5 compares the blocking (including set-up blocking and routing blocking) ratios of QMRA and ARS, with respect to imprecision rate  $\delta$ , when bandwidth

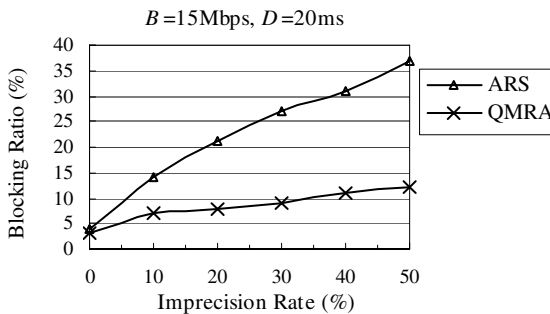


Fig. 5. Blocking ratio vs. imprecision rate

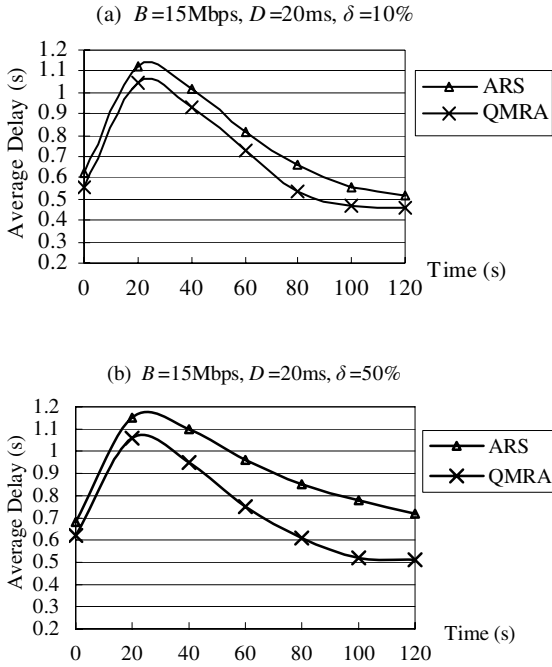


Fig. 6. Average packet delay vs. time

requirement  $B$  and delay requirement  $D$  are fixed. As expected, from this figure it can be seen that our algorithm performs much better than ARS, as the imprecision rate amounts higher. QMRA still has a high success ratio even when the imprecision rate is as high as 50%.

The variation of average packet delay with respect to simulation time is able to indirectly reflect the convergence properties of the algorithms based on ant agents. Seen from Fig. 6(a) and Fig. 6(b), our algorithm converges faster than ARS as the imprecision rate increases. The reason is that QMRA based on a predicting mechanism can more accurately discover the links that satisfy QoS constraints in a network with imprecise state information, where ant agents are applied and the optimization process is usually slow.

## 6 Conclusion

This paper has proposed a novel ant agent-based multicast routing algorithm with bandwidth and delay guarantees, which works for packet-switching networks where the state information is imprecise. In our scheme, an ant uses the probability that a link satisfies QoS constraints and the cost of a path instead of the ant's trip time or age to determine the amount of pheromone to deposit, so that it has a simpler migration process, less control parameters and can tolerate the imprecision of state information.

Furthermore, because of the usage of backward ants, our algorithm can also be applied to asymmetric networks efficiently.

Extensive simulations show our algorithm can achieve low routing blocking ratio, low average packet delay and fast convergence when the network state information is imprecise. Our ongoing work is the further investigation on how to accelerate the convergence of the algorithm and decrease the amount of ant agents needed under the precondition of convergence.

## Acknowledgment

This work is supported by the National Natural Science Foundation of China grant under No. 60172035.

## References

1. Sim, K.M., Sun, W.H.: Ant Colony Optimization for Routing and Load-Balancing: Survey and New Directions. *IEEE Transactions on Systems, Man and Cybernetics, Part A*, Vol. 33, No. 5, 2003, pp. 560-572
2. Bonabeau, E., Dorigo, M. and Theraulaz, G.: Inspiration for Optimization from Social Insect Behavior. *Nature*, London, Vol. 406, No. 6791, Jul. 6, 2000, pp. 39-42
3. Schoonderwoerd, R., Holland, O., Bruten, J. and Rothkrantz, L.: Ant-based Load Balancing in Telecommunications Networks. *Adaptive Behavior*, Vol. 5, No. 2, 1997, pp. 169-207
4. Caro, G.D., Dorigo, M.: Mobile Agents for Adaptive Routing. In *Proc. of the Thirty-First Hawaii International Conference on System Sciences*, Kohala Coast, HI, Jan. 6-9, 1998, vol.7, pp. 74-83
5. Oida, K., Sekido, M.: An Agent-based Routing System for QoS Guarantees. In *Proc. of IEEE International Conference on Systems, Man, and Cybernetics*, Tokyo, Japan, Oct. 12-15, 1999, vol. 3, pp. 833-838
6. Lu, G.Y., Liu, Z.M.: Multicast Routing Based on Ant-Algorithm with Delay and Delay Variation Constraints. In *Proc. of IEEE Asia-Pacific Conference on Circuits and Systems*, Tianjin, China, Dec. 4-6, 2000, pp. 243-246.
7. Guerin, R.A., Orda, A.: QoS Routing in Networks with Inaccurate Information: Theory and Algorithms. *IEEE/ACM Transactions on Networking*, Vol. 7, No. 3, 1999, pp. 350-364
8. Waxman, B.M.: Routing of Multiple Connections. *IEEE Journal on Selected Areas in Communications*, Vol. 6, No. 9, 1998, pp. 1617-1622
9. Ouyang, J., Yan, G.R.: A Multi-group Ant Colony System Algorithm for TSP. In *Proc. of International Conference on Machine Learning and Cybernetics*, Shanghai, China, Aug. 26-29, 2004, vol. 1, pp. 117-121
10. Zecchin, A.C., Simpson, A.R., Maier, H.R. and Nixon, J.B.: Parametric Study for an Ant Algorithm Applied to Water Distribution System Optimization. *IEEE Transactions on Evolutionary Computation*, Vol. 9, No. 2, 2005, pp. 175-191
11. Stützle, T., Dorigo, M.: A Short Convergence Proof for a Class of Ant Colony Optimization Algorithms. *IEEE Transactions on Evolutionary Computation*, Vol. 6, No. 4, 2002, pp. 358-365

# Cactus: A New Constant-Degree and Fault Tolerate P2P Overlay\*

ShuiChao, Huaiming Wang, ZhouPen, and JiaYan

School of Computer, National University of Defense Technology, 410073 Changsha, China

**Abstract.** A fundament tradeoff issue observed by Ratnasamy is a hotspot in designing distribute hash table(DHT) in P2P network. Three constant-degree systems had proposed recently, but the common weakness of them is handle node leaving without inform its neighbors in advance, and optimize the degree and load balance is another question. In this paper, a constant-degree system has proposed named Cactus which based on the 2-tree and CCC hypercube. Its number of neighbor is 6, and the time complexity of key lookup is  $O(d)$  when number of nodes is no more than  $d^*2d$ ,  $d$  is the degree of CCC. In this paper, we will introduce the topology, routing algorithm, node join and leave for Cactus. The experimentations show that Cactus is better in optimizing the degree and load balance, faults tolerate and no worse in other performance compare to other constant-degree system.

## 1 Introduction

Many P2P systems which base on Distribute Hash Table (DHT) had proposed recently. DHT System has two important performance design issue: the number of neighbors (equivalently the size of the routing table) and the network diameter (the number of hops a request needs to travel in the worst case). Ratnasamy[1] had categorized the DHT system into two categories: one has routing table of size  $O(\log_2 n)$  and network diameter is  $O(\log_2 n)$ , e.g. Chord, Tapestry; another has a routing table of size  $O(1)$  and network diameter of  $O(n^{1/d})$ , which includes CAN. One important question proposed by Ratnasamy is whether existing system whose routing table size is  $O(1)$  and network diameter is  $O(\log_2 n)$ . Those systems are also called “Constant degree System”.

Three Constant-degree System had been proposed recently include Koorde[8], Viceroy[7], Cycloid[6]. But as Kaashoed[8] point out that Constant-degree system should been evaluated by five different performance measures: degree, hop count, the degree of fault tolerance, the maintenance overhead, and the degree of load balance. Optimizing one tends to put pressure on the others. Especially, those Constant-degree systems have common weakness is handle node leaving without inform its neighbors in advance, and optimize the degree and load balance is another open question.

In this paper, we proposed one constant degree system called Cactus. It has routing table of size 6 and network diameter is  $d$  when number of system is no more than

---

\* Supported by the National Basic Research Program of under Grant Nos. 2005cb231804.

$d \times 2^d$ . Compare to other constant-degree system, Cactus is better in optimizing the tradeoff between the degree and data load balance, and use a passive child list to handle node accidentally leaving system. Our Simulate result show Cactus's distribute key and lookup load are more evenly than others, and which do well when a fraction of peers leave system without inform its neighbors.

## 2 Related Works

Constant degree system is an important class of the DHT system. There are three constant degree system proposed recently, include Viceroy, Koorde, and Cycloid.

Viceroy[7] maintains a connection graph with a constant-degree logarithmic diameter, approximating a butterfly network. All Viceroy Nodes organized into a ring, and each Viceroy node selects a level at random. So each Peer in level  $l$  has seven links to its neighbors, including pointers to its predecessor and successor pointers in a general ring, pointers to the next and previous nodes in the same level ring, and butterfly pointers to its left, right nodes of level  $l + L$  whose distance roughly  $1/2L$  away, and a down-left edge at a close distance on the ring to level  $l + L$ . Viceroy routing involves three steps: ascending to a level  $l$  node via up links, descending along the down link until a node is reached with no down links, and traversing to the destination via the level ring or ring pointers. Viceroy takes  $O(\log N)$  hops per lookup request.

Koorde[8] combines Chord[3] with de Bruijn graphs. A de Bruijn graph has a node for each binary number of  $b$  bits. A node has two outgoing edges: node  $m$  has an edge to node  $2m \bmod 2^b$  and an edge to node  $2m + 1 \bmod 2^b$ . In other words, a node  $m$  points at the nodes identified by shifting a new low order bit into  $m$  and dropping the high order bit. To look up a key  $k$ , the Koorde routing algorithm must find the successor of  $k$  by walking down the de Bruijn graph. Since the de Bruijn graph is usually incomplete, Koorde simulates the path taken through the complete de Bruijn graph, passing through the immediate real predecessor of each imaginary node on the de Bruijn path.

Cycloid[6] is base on Pastry[4] and CCC[9]. The nodes with the same cubical index are connected by a local cycle, and all the nodes with different cubical indices are connected by a large ring. Each node in Cycloid has 7 neighbors: two neighbors in CCC cycle, a cubic neighbor, two outside leaf neighbors in large ring, and two neighbors which are the first larger and smaller nodes with cyclic index  $k-1 \bmod d$  and their most significant different bit with the current node is no larger than  $k-1$ . Cycloid uses a routing algorithm similar to the one in Pastry, and has a much shorter path length per lookup request in the average case than Viceroy and Koorde.

## 3 Cactus

A  $d$ -dimensional CCC graph is a  $d$ -dimensional cube with replacement of each vertex by a cycle of  $d$  nodes. Each node is represented by a pair of indices  $(k, a_{d-1}a_{d-2} \dots a_0)$ , where  $k$  is a cyclic index and  $a_{d-1}a_{d-2} \dots a_0$  is a cubical index. The cyclic index is an integer, ranging from 0 to  $d - 1$  and the cubical index is a binary number between 0 and  $2^d - 1$ . Each node in CCC graph has three neighbors: one cubical neighbor whose

coordinate are  $(k, a_{d-1}a_{d-2} \dots \bar{a}_k \dots a_0)$ ; two cyclic neighbors, one's coordinate are  $((k-1) \bmod d, a_{d-1}a_{d-2} \dots a_0)$ , and another is  $((k+1) \bmod d, a_{d-1}a_{d-2} \dots a_0)$ . The network diameter in CCC structure is  $2d$ , but message should be cycled in network when some nodes depart from system.

A 2-tree had been created to solve this problem in Cactus. The figure 1(a) shows this tree which the root's coordinate are  $\text{root}(0 \dots 01)$ ; each node  $(0 \dots 01a_k \dots a_0)$  who is not leaf in tree has two children, the index of the left child are  $(0 \dots 011a_k \dots a_0)$ , and the index of the right child is  $(0 \dots 010a_k \dots a_0)$ . For example the vertex  $a(0110)$  have a farther  $b(0010)$ , left child  $c(1110)$ , and right child  $d(1010)$ .

Now let every vertex  $(k, a_{d-1}a_{d-2} \dots a_0)$  in CCC has a father and two children whose cubic index's relationship is as same as in 2-tree, and cyclic index is numerically closet to  $k$ . Furthermore, when nodes whose cubic index is  $a_{d-1}a_{d-2} \dots \bar{a}_k \dots a_0$  do not participant the system, the cubical neighbor is the father of that node. For example, if  $a(0, 0110)$ 's cubic neighbor  $(x, 0111)$  does not participant system, the node  $(0, 0011)$  is  $a$ 's cubic neighbor.

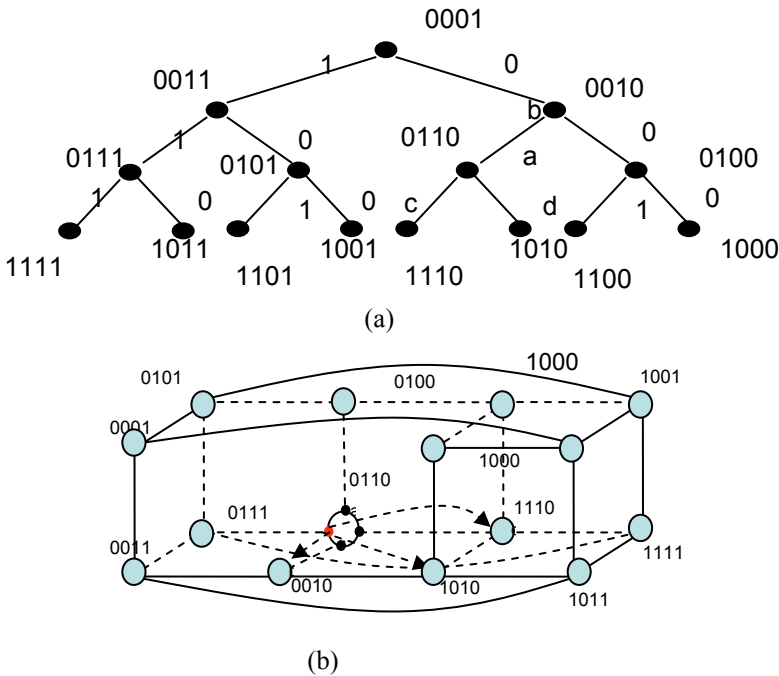


Fig. 1. Topology of Cactus

The figure 1(b) plots 4-dimension Cactus. Each node in Cactus has 6 neighbors: two neighbors in CCC cycle, one cubic neighbor, one father and two children in tree. For example, the node  $a(0,0110)$  has 6 neighbors which present in table 1:



**Table 1.** The neighbor table of node (0,0110)

ID	(0, 0110)
cubic neighbor	(0, 0111)
cyclic neighbor	(1, 0110)
	(3, 0110)
father neighbor	(0, 0010)
left child	(0, 1110)
right child	(0, 1010)

### 3.1 Cactus Routing Algorithm

The tree routing algorithm or CCC routing algorithm can be used in Cactus. But tree routing algorithm lead more load in upper level peers than lower level peers, and CCC routing algorithm is not tolerate the circumstance that peer join or depart the system frequently. So we use a routing algorithm to solve those problems.

The idea of this algorithm is peer forwards message using the CCC routing algorithm firstly, but peer will use the tree algorithm when the destination peer's cubic index is the ancestor or child of current peer. So we call this algorithm the Hypercube first search, or HFS. Suppose the coordinate of current peer are  $p(k_p, p_{d-1}, p_{d-2}, \dots, p_0)$ , and destination peer's coordinate are  $a(k_a, a_{d-1}, a_{d-2}, \dots, a_0)$ . The algorithm is presented as follow which has three phases: the cubic routing, the tree routing and cyclic routing.

- (1) if  $a$  is the ancestor or child of  $p$ , then (3).  
     else if  $p$ 's cubic index is as same as  $a$ 's cubic index, then (4).
- (2) The cubic routing:  $p$  calculate the minimal Difference Bit(MDB) between the  $p$  and  $a$ . The peer forward message to cubic neighbor when  $k_p = \text{MDB}$ ; or peer forwards the message to success peer in cyclic when  $\text{MDB} > k_p$  and  $\text{MDB} \geq \text{success's cyclic index}$ ; or the peer forwards the message to subsequence peer in cyclic when  $\text{MDB} > k_p$  and  $\text{MDB} \geq \text{subsequence's cyclic index}$ , or forwards to father else.
- (3) The tree routing:  $p$  forwards message to farther when  $a$  is the ancestor of  $p$  or to child when  $a$  is child of  $p$ , or forwards message to success or subsequence peer in cycle when father peer disappear accidentally.
- (4) The cyclic routing:  $p$  forward message to success peer or subsequence peer decide by whose cyclic index is close to destination peer.

For example, just like 4-dimension Cactus presented in figure 1. The requester  $a(0,0110)$  want to search a peer  $m(3,1101)$ . The MDB between the requester and destination peer is 0. So message firstly forwards to (0, 0111) because the  $a$ 's cyclic index=MDB. At next time, the  $\text{MDB}=1$  and cyclic index<MDB, so the message forwards to (1, 0111). After node(1, 0111) found the MDB between it and destination peer is 1, and cyclic index=MDB, it forwards message to (1,0101). At this time, node(1,0101) is the farther of the destination node. So the algorithm goes out the hypercube phrase and starts the tree routing phrase. The message is forwarded to (1, 1101). As the destination peer is in the same cycle with current peer, the algorithm goes into the cyclic routing. The message forwards to destination peer(3,0101) through the subsequence peer in the cycle. There are 6 steps when the requester sends message to destination.

The weakness of CCC structure is the destination peer may not be reached or some message may be cycled when some peer leave the system. In Cactus, we can prove its convergence and reachability by theorem 1:

**Theorem 1.** The length of routing path is no more than  $2m$  when message forwards in  $m$  level of tree, and message doesn't forward to peer at  $k(k>m)$  level of tree when route is in the hypercube phrase.

Prove: The current peer in  $m$  level of tree face three case:

- (1) The destination peer is the child of current peer if  $MDB > m$ . it is the end of hypercube phrase.
- (2) if  $MDB < m$ , then
  - (a) The message is forwarded to hypercube neighbor when cyclic index =  $MDB$ . Its neighbor is also in  $m$  level of the tree by character 1.
  - (b) Otherwise, the message is forwarded to success peer or subsequence peer.
- (3) Message is forwarded to father peer when  $MDB = m$ .

Only the (1) cause the message forwards to lower level of the tree and goes into the tree routing phrase. So message should not been forward to lower level peer in hypercube phrase. At (2)(3), each step reduces the distance between the destination peer and requester. So the length of routing path is no more than  $2m$ .

End.

For each length of three routing phrase of HFS algorithm is  $O(d)$ , the length of routing path is  $O(d)$ . In fact, the longest path in Cactus is  $(5/2)*d$ , and mean of routing path is  $(5/4)*d_c$ .

### 3.2 Key Location

For a given key, the cyclic index of its mapped node is set to its hash value modulated by  $d$  and the cubical index is set to the hash value divided by  $d$ . If peer want to publish a key, the consistent hashing will map key to node  $k$  and a lookup request can be redirected in  $O(d)$  steps following the routing algorithm. If the target node of a key's ID( $k_a, a_{d-1} \dots a_0$ ) is not a participant, two different result should been return: one is the node whose cyclic index is first numerically closest to  $k_a$  and cubic index is same as  $a_{d-1} \dots a_0$ ; or return the node who is the ancestor of node( $k_a, a_{d-1} \dots a_0$ ) and cyclic index is closest to  $k_a$ . In the case of two nodes with the same distance to the  $k_a$ , the key's successor will be responsible for storing this key. For example, the node(1,1001) is responsible for storing key(0,1001) when node(0,1001) doesn't existing in system; or node(1,0101) is responsible for storing key when no node existing in system whose cubic index is 1001.

### 3.3 Self-organization

In practice, Cactus needs to deal with nodes joining the system and with nodes that fail or leave voluntarily. This section describes how Cactus handles these situations.

### 3.3.1 Node Joins

As all the P2P system, a new node should get a live node firstly. It select a coordinate  $m(k_m, m_{d-1} \dots m_0)$ , and send joining message by live node. The message should forwards to a node  $c(k_c, c_{d-1} \dots c_0)$  by routing algorithm, two cases are considered:

- 1) If  $c$  is the father of  $m$ , then  $a$  is the first node in its cycle. So  $m$  creates its neighbor table, which two children are  $c$ 's children, father is  $c$ , and two neighbors in cycle is null. Then  $m$  send a search request for  $m$ 's cubic neighbor, and record the result in neighbor table. Meanwhile,  $c$  should broadcast the child joining message in its cycle for every node in cycle can update its child entry.
- 2) If  $c$  is the successor or predecessor of  $m$ , then  $m$  and  $c$  is in same cycle. Then  $c$  send three search message for  $m$ 's cubic neighbor, father and child, and  $m$  records results in his neighbor table. Then  $m$  broadcasts an update message around the cycles which father and children node exists. The node who receives the message should update its neighbor table when  $m$ 's cyclic index is numeric closest to it.

There are  $O(d)+3d$  messages for a new node join to system. The number of message is more than Koorde or Chord when new node joins system, but which insure the fault toleration of system as we will discuss in next section.

### 3.3.2 Node Departure

Before node departs from system, it should notify all neighbors and inform them its neighbor table. Two cases should be considered especially. First, father and children nodes should broadcast the node leaving message one by one in their cycle for every node in cycle can update its neighbor table. Second, the cubic neighbor should point to departed node's father when departed node is the only node in its cycle. In those scenes, the longest message path is  $d$  and number of messages is no more than  $3d+3$ .

### 3.3.3 Fault Tolerance

The fault tolerance means that fraction of the nodes can fail without eliminating data or preventing successful routing. Only a little fraction of the nodes depart from system accidentally cause the constant-degree system trends to remain connected with low probability and routing inefficiently[6]. The reason is that some hot spots had existing in those systems such as Cycloid's primary node of a cycle, Koorde's first de Burijn node, and father node in Viceroy. But no such hot spot had existed in Cactus. In hypercube phrase, the message forwards to father when cubic neighbor does not exist. In tree routing phrase, the successor or predecessor help message forwards to correct node when father or child connection is lost. So Cactus can tolerate a little fraction of node leave the system accidentally without other mechanism. As described in Section 4.5, the Cactus achieves 10% location failure when 20% nodes leave the system without inform its neighbors.

But Kaashoed had proved that in order for a network to stay connected with constant probability when all nodes fail with probability  $1/2$ , some nodes must have degree  $O(\log n)$ . So the koorde using successor list like chord and Viceroy using Bucket to resolve those problem. Cycloid had to solve this problem in future work. Those solutions had destroyed the optimality of constant degree which node must maintain the correction of the backup node list. In other hand, the message for system forward

message to next step is  $O(\log N)$  when every node fail with probability  $1/2$ . How to resolve those problems and keeping the constant degree is an open question.

That problem is solved by Cactus using a dynamic mechanism. The idea is that system let a child node help father to found his successor or predecessor which departs from system accidentally. As node joining algorithm discussed above, the new node joining message should be broadcasted in father node's cycle when node joins the system. So non-leaf node can maintain a children table to record all nodes in his child's cycle. Please notice that node only maintain this table passively. In other word, node does not use some machine to insure the correction of children table. So this machine does not aggravate burden of network and system. When node's successor or predecessor is fail, it will send a repair message to all nodes which have an entry in children table. The node who receives the repair message should record requester to a list, and then returns the list to requester after an interval. After an interval, the requester knows who is living in its cycle and its child also knows which father node is living. Using this mechanism, each non-leaf node has a children table, but it can not maintain it actively. It does not breach the principle of constant-degree system that node has  $O(1)$  neighbors for decrease the system and network's burden. Meanwhile, the hops for repair message are 2, and the lookup hop should not increase when using this machine because it does not influence the routing algorithm.

## 4 Performance Evaluation

In this section, we will compare Cactus to other two constant-degree system: Koorde and Cycloid. Viceroy is not consider because the [6] had proved that Viceroy's performance is worse than Koorde and Cycloid. CAN is also considered for comparing the constant-degree system to non-constant-degree system. Four performance have been considered which including degree in terms of the number of neighbors to be connected, hop count per lookup request, degree of load balance, degree of fault tolerance. In our experiments, Koorde have 7 neighbors each node including one de Bruijn node, three successors and three predecessors; and the neighbor table size is 7 in Cycloid and Viceroy yet. The simulation results due to Cycloid, Koorde, and Viceroy were verified with those in their original publications.

### 4.1 Lookup Efficiency

All constant-degree systems have  $O(\log N)$  hops per lookup request with  $O(1)$  neighbors. But different topology cause different lookup efficiency. In our experiments, the sizes of nodes increases from 128 to 16k vary 2 times each experiment. Under different scale, each node sends out 100 lookup requests randomly. Figure 2 show the mean of the measured path lengths of the lookup requests due to various DHT routing algorithms with respect to the node scale.

We can see that mean length of Cycloid's lookup path is smaller than Cactus. The import reason is that the leaf in Cactus has no child item in neighbor table but there is no null pointer in Cycloid's neighbor table. Compare to the mean of out-degree is 7 in Cycloid; the mean of out-degree in Cactus is 4. So the Cycloid achieve the almost same lookup efficiency with Cycloid but smaller size of out-degree.

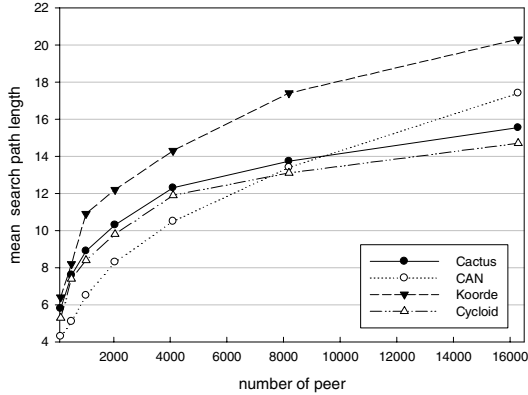


Fig. 2. Lookup efficiency for different DHT

## 4.2 Data Load Balance

How evenly the keys distribute among nodes in constant-degree system is an open question[8]. We use two experiments to compare the different performance of data load balance between four constant-degree systems. One is under situation that the nodes are dense; another is under situation that the nodes are sparse. 2000 nodes with 2048 ID space are adopted in first experiment. The number of keys varies from 10000 to 100000 in increments of 1000. Fig 3(a) plots the mean, the 1st and 99th percentiles of the number of assigned keys per node. The key distribute in Cactus is same as Cycloid because they using hypercube as their topology. The experiment also shows that Koorde have same performance with Cactus and Cycloid when the node is dense.

But the performance is different when the node is spare. We do the second experiment with 1000 nodes in 10240 ID space. Other condition is same as the first experiment. The fig 3(b) shows the mean, the 1st and 99th percentiles of the number of assigned keys per node. The Cactus lead better data balance performance than Cycloid and Koorde in spare network. The reason is that there maybe a continue ID space not existing in node sparse situation. In Koorde, those keys will store in its successor. Cycloid achieves better load balance by storing the key in its numerically closest node; that is the keys between a node's counter-clockwise neighbor and itself will be allocated to that neighbor or the node itself rather than to itself totally. Similar to Cycloid, Cactus stores those key in node whose cyclic index is numerically closest to key and cubic index is same as key, or store those key to his father when no cycle neighbor existing. For example, suppose the ID from  $(x, 10000111)$  to  $(x, 1000001)$ ,  $8*6=48$  nodes, are not existing in network. Koorde store those key in the node(10000000), and Cycloid stores those keys in  $(x, 01111111)$  or  $(x, 10000000)$  which decide by who is numerically closest node. But Cactus will stores those keys among the node  $(x, 0100001)$ ,  $(x, 01000010)$ ,  $(x, 01000011)$ ,  $(x, 01000100)$ ,  $(x, 01000111)$ ,  $(x, 01000110)$ . In other word, Cactus distributes keys among more nodes in node sparse situation, and shows better performance of data load balance than Koorde and Cycloid.

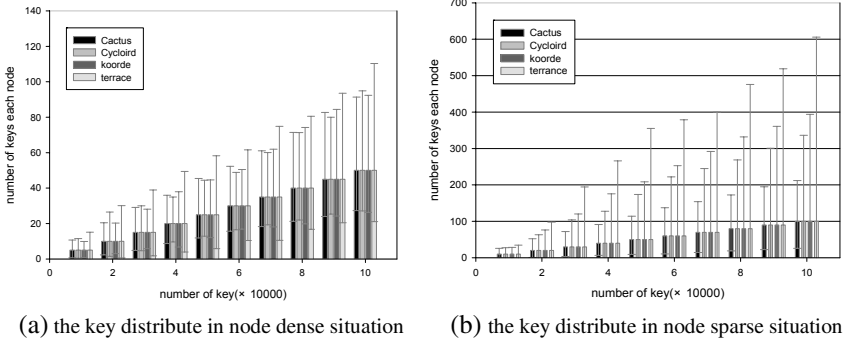


Fig. 3. The key distribute due to different DHTs

### 4.3 Routing Load Balance

Routing load balance means how evenly the search request among the nodes. The systems base on tree topology lead more load on upper level node than lower level node. The Cactus use tree topology and whether it shows that character is a question. In next experiment, we compare Cactus with PGrid and Terrace which base on the tree topology. Figure4(a) shows the different between them when 2048 nodes existing in network. In PGrid, the load of the upper node is obviously higher than the lower node. Terrace using a “random link” ease the upper node’s load. In Cactus, the message has initiative forward to hypercube neighbor which cause the message transmits on same layer of requester when nodes are dense. Therefore, the average load of Cactus each layer almost has no difference.

In hypercube routing phrase, the message in Cactus firstly forwards to node with small cyclic index that leads light load for node with big cyclic index. The same situation is exiting in Cycloid which nodes with small cyclic index have light workload than big cyclic index. So we do an experiment to compare the workload of node with

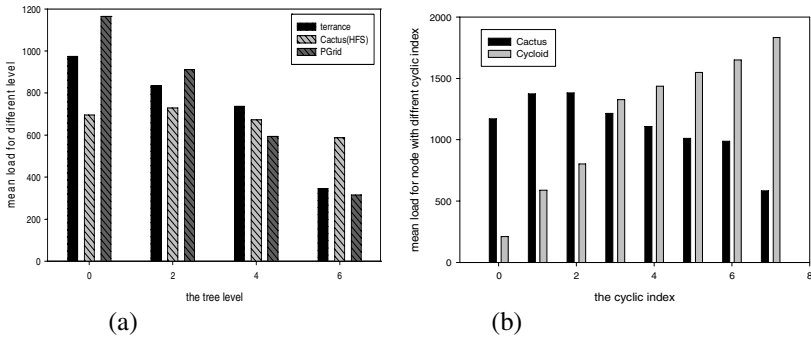


Fig. 4. The routing load distribute among the node due to different DHT

different cyclic index. Fig 4(b) shows the routing load between the Cycloid and Cactus. In ascending phase of Cycloid, the message forwards to the node with biggest cyclic index at first time. So the node with biggest cyclic index become the hot spot especially the nodes is dense. Otherwise, there is no hot spot in Cactus because the node ID bits change one by one and start at requester’s cyclic index.

### 4.4 Fault Tolerate

In this section, we evaluate the impact of massive node failures and/or departures on the performance of various DHTs, and on their capability to performing correct look-ups without stabilization. We do two experiments; one assumes that node departures are graceful that a node informs its relatives before its departure. Another assumes that node departures accidentally.

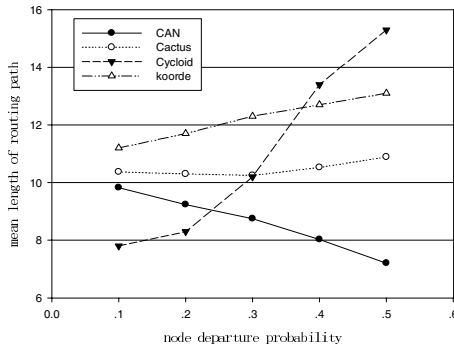


Fig. 5. Path length with different node departure probability

In first experiment, 2048 nodes are existed in system. Once the system become stable, each node departs from system with probability ranging form 0.1 to 0.5. After departure occurs, we perform 1000 time lookup with random source and destination. Fig 5 plots the result of experiment. From the picture, we can see the length of lookup path increase for Cycloid due to the increasing of the number of timeouts as the p increases. When a departed node is met, the leaf sets have to be returned to for the next node. Therefore, the path length increases. The path length doesn’t increase in Cactus and Koorde as the p increases. It is because timeout event only occurs when cubic neighbor depart from system accidentally in Cactus, and the message will forward to father node at this point. Route increase one step only at hypercube routing phrase but also decrease one step at tree routing phrase. So the path length keeps stable as p increase in Cactus.

To achieve good fault tolerate performance, P2P usually maintain a backup nodes list such as “success list” in Chord or “Bucket” in Viceroy. In addition to maintain the connection with neighbors, those machines lead system actively maintain the connection with other node for fault toleration. It aggravated the burden of network. How well the passive fault toleration machine work in Cactus is question we must answer. The experiment let node leave the system with probability range form 0.1 to 0.4, and node does

not inform its neighbors. The figure 6 plots the result of experiment. Compare to 40% location failure in Chord when the random failure scale is 40% with node departure probability 0.4, the location failure ratio is about 10% in Cactus with tolerate mechanism. The reason is node actively repairs the connection with living node in Cactus when it found successor or predecessor is losing. Nodes in child's cycle and father's cycle help node resume the connection with correct neighbor. The request message waiting at current node until correct route resumes, and continues its travel in Cactus.

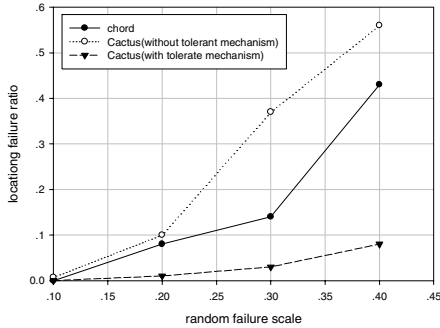


Fig. 6. Location failure ratio with different node failure scale

## 5 Conclusion

In this paper, we propose a new constant-degree P2P overlay named Cactus. Its neighbor table size is 6, and the mean length of lookup path is  $O(d)$  when the number of nodes is no more than  $d \cdot 2^d$ . It is similar to Cycloid which base on Pastry and CCC, but Cactus base on CCC and tree. Compare to Cycloid, Koorde, and Viceroy, Cactus Optimize neighbor table size with lookup hop, data load balance, routing load balance, and fault tolerate:

- (1) Cactus achieves higher degree of data load balance than Koorde and Cycloid for spare network, and higher degree of routing load balance than Cycloid.
- (2) Cactus keeps correction and efficiency when node joins and departs system frequently. Cycloid increases length of routing path and Koorde increases failure times when some nodes depart from system.
- (3) Cactus uses passive tolerate mechanism for achieve the good fault tolerate performance and doesn't aggravate the burden of network.

## References

- [1] S. Ratnasamy, S. Shenker and I. Stoica. Routing Algorithms for DHTs: Some Open Questions. In Proceedings of 1st International Workshop on Peer-to-Peer Systems (IPTPS'02). Cambridge, MA, USA, March 2002.
- [2] Ratnasamy S., Fancis P., Handley M, A scalable content-addressable network. Proceedings of ACM SIGCOMM, San Diego, CA,USA,2001



- [3] Stoica I. Chord: A scalable peer-to-peer lookup protocol for Internet applications. Proceedings of SIGCOMM, San Diego, CA USA, 2001
- [4] Rowstron A., Druschel P, Pastry: Scalable, decentralized object location and routing for large-scale peer-to-peer systems. Proceedings of the 18th IFIP/ACM International conference on Distributed Systems Platforms, Heidelberg , Germany, 2001
- [5] Zhao B.Y, Tapestry: A resilient global-scale overlay for service deployment. IEEE Journal on Selected Areas in communications, 2004, 22(1)
- [6] Shen H.Y, Xu C.Z., Chen G. Cycloid: A new constant-degree and lookup efficient P2P overlay network. Proceedings of International Parallel and Distributed Symposium (IPDPS'04), Santa Fe, USA, 2004
- [7] Malkhi d., Ratajczak D. Viceroy: A scalable and dynamic emulation of the butterfly. Proceedings of Principles of Distributed Computing (PODC 2002), Monterey, CA, USA, 2002
- [8] Kaashoek M.F., Karger R. Koorde: A simple degree optimal distributed hash table. Proceeding of the 2nd International Workshop on P2P Systems (IIPS'03), Berkeley, CA, USA, 2003
- [9] Perparata F.P, Vuillemin J, The cube-connected cycles: A versatile network for parallel computation. Communications of the ACM, 1981 24(5):300~309
- [10] Dou Wen, Jia Yan, Song Wen-qiang, Zou Peng, A P2P Approach for Global Computing, International Parallel and Distributed Processing Symposium (IPDPS'03), IEEE Computer Society, Nice, France , 2003.4, pp248-255
- [11] Mario Schlosser, Michael Sintek, Stefan Decker, Wolfgang Nejdl, HyperCuP—Hypercubes, Ontologies and Efficient Search on P2P Networks. AP2PC 2002, Bologna, Italy
- [12] K. Aberer, P. Cudre-Mauroux, A. Datta, Z. Despotovic, M. Hauswirth, M. Puceva and R. Schmidt. P-Grid: A Self-organizing Structured P2P System. ACM SIGMOD Record, 2003, 32(3).
- [13] Qutaibah M., Malluhi and magdy A., The Hierarchical Hypercube : A new Interconnection Topology for Massively parallel systems. IEEE Trans. Parallel Distrib. Syst. 5(1): 17-30, 1994
- [14] S. Ratnasamy, P. Francis, M. Handley, R. Karp and S. Shenker. A Scalable Content-Addressable Network. In Proceedings of ACM SIGCOMM. San Diego, California, USA, 2001.
- [15] K. Gummadi, R. Gummadi, S. Gribble, S. Ratnasamy, S. Schenker and I. Stoica. The Impact of DHT Routing Geometry on Resilience and Proximity. In Proceedings of ACM SIGCOMM: 381-394. Karlsruhe, Germany, Sep. 2003: ACM Press.

# MPSS: A Multi-agents Based P2P-SIP Real Time Stream Sharing System

DeGuo Yang, Hui Wang, CuiRong Wang, and Yuan Gao

School of Information Science & Engineering, Northeastern University,  
110004 Shenyang, China  
yangdeguo2002@163.com

**Abstract.** P2P systems have high scalability, robustness and fault tolerance because of its no centralized server and the network self-organized itself. As the standard protocol for VoIP and NGN, SIP is a general protocol for establishing and controlling multimedia session. In this paper, we proposed a peer to peer based realtime streaming media sharing system using SIP mechanism, which uses Chord as the underlying connection architecture. We add the media agents sever to support real time streaming media downloading. We have advanced peer and piece selection algorithm to solve the real time media stream transmission. The MPSS system architecture supports user registration, media resource location, media streaming session establishment, and realtime media streaming playback online. At last, we give a simulation and test result.

## 1 Introduction

With the development mobile wireless communication, the requirement of multimedia on mobile telephone and other handset has increased. Because the store space and bandwidth of this kind of equipment is limited, the suitable method is getting the multimedia file on line, playing while downloading. The other users of PC also prefer to watch the media file without waiting until the file is downloaded. It is necessary to develop a real time media system to transmit the media among the users.

Peer-to-peer (P2P) system is inherently scalable and reliable because of the lack of a single point of failure [1]. P2P system, in the purest form, has no concept of servers. All participants are peers and communicate in distributed to achieve a certain objective. Although peer-to-peer media streaming system has received more and more attention, general peer-to-peer system only offer the function of file sharing, and the mode could not satisfy the demand of real time media stream transport. On the other hand, Session Initiation Protocol(SIP)[2] is an IETF standard for Voice over IP (VoIP). As SIP media application have increased in popularity, situation have emerged where centralized server are either inconvenient or undesirable. But as the IETF standard, SIP application is popular in communications systems.

In this paper, we design the MPSS real time media sharing system. MPSS combines the SIP family of IETF standards for VoIP(Voice Over Internet Protocol) with the P2P mechanism. The result is a standard based, decentralized

online media sharing system. Our design allows for compatibility with and reuse of existing SIP network elements. The P2P overlay is built using a SIP agents through exchanging SIP messages, which are typically well supported by existing firewalls, and therefore retain compatibility with the current network infrastructure. MPSS uses advanced media stream piece selection algorithm and peer selection mechanism to solve real time media transport problem. To ensure the reliability of the real time media stream, we adopt some media agent server to store and buffer some important media blocks.

The main contributes of this work are:

Creating a fully distributed, SIP based real-time multimedia file sharing system.

In the design of system, using advanced media streaming hand out mechanism to support playback the media while downloading.

Proposing a media transmission agents scheme to improve the effect of media downloading.

## 2 Background and Related Work

### 2.1 P2P and SIP

The fundamental principle behind peer-to-peer (P2P) architectures is that each and every member of the network has equal importance in the transactions that take place on the network, and that these nodes communicate with each other to accomplish tasks. The best known use of P2P technology is file sharing system. But the P2P always face the problem of the instability brought by the peer impolite leave or shut down, and the ordinary peer-to-peer file sharing system is not suitable for the real time media transport for the lack of optimized peers and pieces mechanism for the real time stream media. Such as BitTorrent[5], although its peers selection policy effectively prevent the free rider[4], it limits the overall throughput and reduces the peer band-width utilization rate of the system, which is a important defect for the real time stream transport. Another peer-to-peer live multimedia stream system is Chainsaw[6]. Although Chainsaw is pull-based and effectively utilizes the source uplink bandwidth, but it has not the pieces optimal selection policy, and is not scalable.

SIP has emerged as the standard protocol for VoIP. SIP supports the initiation, modification, and termination of media sessions between user agents. Although some systems use other standards or proprietary mechanisms such as H.323, the majority of new VoIP development uses SIP. SIP with SIMPLE extensions is also extensively used by existing IM clients, for example in Microsoft's MSN messenger. Using the open SIP standard helps us meet our requirement for compatibility and allow the system to reuse open source stacks and the application can be integrated with the tremendous number of existing SIP terminal [1]. The applications of SIP are increasingly popular in communications systems for private, corporate, and academic purpose. Because of the character of the peer system, the peers with important information shut off from the system will

bring about inevitable loss for the other peer. It is a fatal defect to some applications which have high quality for the communication link and streaming media delivery. Some service agent servers become necessary in those applications.

## 2.2 Distributed Hash Table (DHT) Systems: Chord

To improve the peer and resource locating efficiency, most P2P systems locate resource by a Distributed Hash Table (DHT). The Chord[7] is one particular DHT algorithm. In this system, every node has a Node-ID which is got by hashing the IP address and port of the node; and every resource has a Resource-ID, which is obtained by hashing some keyword or value that uniquely identifies the resource. The node is responsible for storing all resources that have Resource-IDs near the node's Node ID. Chord uses a ring-type structure for the nodes in the overlay. In this structure, if the hash has  $2^n$  bits in the range, each node will keep a finger table of pointers to at most  $n$  other nodes. The  $i$ th entry in the finger table contains a pointer to a node at least  $2^i$  units away in the hash space. These highest finger table entries thus point to a range  $1/2$  of the way across the hash space, the next  $1/4$ , and so on, and the smallest entry points to a range only 1 away in the hash space. Searching in Chord is accomplished by sending messages to the node in the finger table that is closest to the destination address. That neighbor will have finer resolution detail about the area and can route the message closer to the desired node. This process is repeated until the message reaches the node responsible for the destination.

## 2.3 Real-Time Media Streaming Transport

Although there have been significant research efforts in peer-to-peer systems during the past two years, but peer-to-peer media real time streaming system has received less attention. The character of its play-while-downloading mode is more attractive than the ordinary one. The reference[3] proposed the algorithms  $OTS_{p2p}$ . The algorithm  $OTS_{p2p}$  offers an optimized piece choice policy which computes the media data assignment for each peer-to-peer streaming session. The assignment will lead to the minimum buffering delay experienced by the requesting peer. After computing the media data assignment, it will initiate the peer-to-peer streaming session by notifying each participating supplying peer of the corresponding assignment. In real time streaming media transport, the crucial problem is that the system should guarantee every peer can get the needed media file piece in time, while in most paper, the problem is not mentioned.

## 3 MPSS System Design

Our design has no central servers and nodes communicate directly with one another, there are some differences between our approach and traditional SIP. A peer connects to a few other peers in the overlay network and use SIP message to search file message and locate nodes. Node act both as UA(user agent) and

Proxy simultaneously. When it gets message and searches file it acts as UA; and when it maintains and records the file message and responses the request node, it acts as proxies and replace the functionality of SIP registrars and proxies. Each node is responsible for those roles for some portion of the overlay.

### 3.1 Node Structure

Nodes are organized in a DHT based on Chord. To provide rapid resource location, MPSS maintains only 16 finger table entries for routing. All messages for maintaining the DHT, registering users, locating resources, and establishing sessions are SIP messages. SIP is not designed with supporting P2P, but we have not needed to add new methods to SIP to implement P2P functionality but only define a few new SIP headers. We use SDP to describe the media file information, such as its length, name, hash information and so on.

We use the SIP REGISTER message to pass the overlay information between nodes, as well as the original use of sending user location information to a registrar (peer). If a new node joins the system, a number of REGISTER messages should be exchanged. At first, the node is assigned a Node-ID created by hashing (using SHA-1) the real IP address and appending a port. Then send a REGISTER message to a node which is responsible for the region, and insert itself into the overlay. Once the new node has joined the overlay, it will be responsible for storing information (user registrations) associated with the portion of the overlay mapped to that calculated Node-ID, and transfer the information for the region from the node presently storing that data which should belong to the new joining node. We use INVITE message with SDP information to request suitable user to transfer needed peer information and media pieces.

### 3.2 File Publishing and Pieces Distribution

To start a media file deployment, a static file is stored on a node according to Chord architecture. The file contains information about the file, its length, name, and hashing information and the information which nodes are downloading it and downloading what part of it, and the downloaders' IP addresses and ports. If a node wants to publish content, it hashes the key word of the content, and finds the node which should be responsible for the content according to the Chord peer-to-peer structure. The node which is responsible for the content will store the static file.

When a file is published on the overlay network, the node which stores the file information will acts as a tracker. The tracker is a manager of the file on the node, it receives from all downloaders and giving them a random lists of peers. The downloaders periodically checking in with the tracker to keep it informed of their progress, and the tracker offers the information of the file information.

For the purposes of playing while downloading with best effort, the seed file is par-titioned into fixed-size pieces which are all the same length except for possibly the last one which may be truncated. Piece length is almost always a power of two, most commonly  $2^8 = 256$ Kbytes. The seed computes an SHA1 hash for each piece and distributes these hash values to each receiver so that

the latter can verify the integrity of the pieces it receives later on. Each piece is further divided into blocks, typically of size 16Kbytes. A receiver can download blocks within the same piece from multiple peers simultaneously. However, a node can relay blocks of a piece to other nodes only after it receives the entire piece and successfully verifies its integrity.

### 3.3 Peer Strategy and Piece Selection

Once a node wants to download a file, it will hash the resource file key word, and get a file ID. According to the file ID, The node searches its finger table and sends the request information to the overlay network. If the file is not existed, the node whose node ID is just equal or follows the file ID will return the response that the request file is not exist. If the File does exist, the tracker on the node who is responsible the file will return the request node a file information including the nodes random lists table with which the request node can download the file from. Then the request node chooses the node from the list in some policy to download the file pieces which it needs. Another method to get the need file information is to enter the specified web page with the available resource information, and click the hyperlink on the web page, then, download needed P2P file store information and file pieces.

**Peer Strategy.** Peer strategy has a great impact on local performance. To join a session, a node first contacts the tracker to obtain a list of randomly chosen peers that are currently in the session. Then it establishes new socket connections with these peers until the number of connected peers reaches some threshold, in this system, the threshold is 40. In the mean while, it can start accepting connection requests from other peers, but will stop accepting new connection requests when the number of connected peer reaches another threshold, currently 80. A client will asks the tracker for a new list of peers either every some timegenerally 30 minutes, or when the number of neighboring nodes reaches the minimum, currently 20.

Given a set of file block from its peers, a node needs to determine which request to service and which to ignore. The system should employ a fairness policy to prevent free riders[4]. Especially in a real time streaming media system, we should not only consider the free riders but also the system throughout and make fully use of the bandwidth of all users. It is obvious that the admission which favors higher class requesting peer will lead to faster amplification of the peer-to-peer system capacity, and will ultimately benefit requesting peers of all class. This will create an incentive to encourage requesting peers to contribute its truly available outbound to the system.

In constructing the multimedia application, many factors need to be considered, such as delay, bandwidth, loss rate and so on. To use a single metric as the indicator in peer selection is obviously not sufficient for supporting real time multimedia applications. Thus, we choose peers based on both delay and available bandwidth measurement. Generally, the peers' download bandwidth will be higher than upload bandwidth, so we only take the upload bandwidth into consideration.

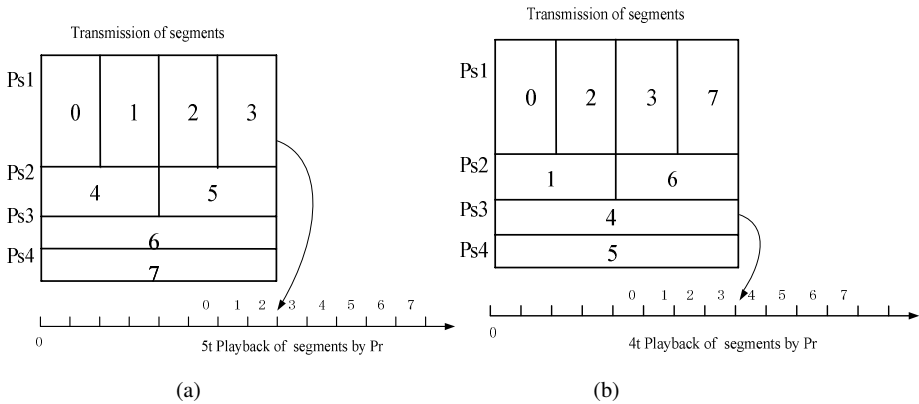
In this design, we use the mechanism of *measuredvalue* of the node to dynamically adjust the parameter  $p$  to trade off the relationship between bandwidth and latency in practice. We defined the capability of a peer as the *measuredvalue* between them, and the *measuredvalue* is calculated according to the following method (the unit of delay is  $s$  and the unit of bandwidth is  $M$ ):

$measuredvalue = bandwidthp + (1 - delay/10)(1 - p)$ ; ( $p$  is a number between 0 and 1, it can be adjusted in practice, when the delay is over 1s, the value of  $p$  is 0, in this system the  $p$  is set 0.5).

To a peer which want to download the media file, It is supposed that there are  $N$  linked nodes have the needed file piece. The peer will calculate the *measuredvalue* of the  $N$  neighbor nodes, and select the first 5 peers with higher *measuredvalue* than the rest as its selected peer to download and upload peers.

As is well known, the Internet is changing dynamically. The end-to-end delay and the available bandwidth between conference members are also not static. Thus, we need a mechanism to duly measure these two indicators in order to build efficient link. The peer members periodically probe the selected to obtain the end-to-end delay. The probing requests are also used for member failure detection: if member A does not receive a probing message from member B over a period of time, it diagnoses B as having a network failure. This is also referred to as the heart-beat mechanism. The heart-beat (probing) interval is 5 seconds, and the failure diagnostic threshold is 20 seconds. We adopt the Packet-Pair[8] method to measure the available bandwidth. In our current implementation, each member probes others in turn with a pair of 1.5 KB UDP packets at an interval of 10 seconds. Thus, in a typical five member conference, the available bandwidth between every two members is determined every 40 seconds. For every 5 seconds, each member will broadcast its updated measurements in the conference. It is obvious that the overall overhead is less than 4 kbps, which is affordable by most Internet users.

**Pieces Selection.** Once a peer establishes connections with its potential peers, it needs to determine which pieces to download from which peers, based on



**Fig. 1.** Different assignment leads to different buffering delays

the knowledge of the set of file pieces available in all its peers. As shown in Figure 1, different assignment policy leads to different buffering delays. In the Figure 1, the requesting peer is  $P_r$  and the playback rate of the media data is  $R_0$ . The supplying peers are  $P_{s1}$ ,  $P_{s2}$ ,  $P_{s3}$ ,  $P_{s4}$ . with out-bound bandwidth of  $R_0/2$ ,  $R_0/4$ ,  $R_0/8$ , and  $R_0/8$  respectively. In assignment(a)  $P_{s1}$  is assigned media data segments  $8k$ ,  $8k + 1$ ,  $8k + 2$ ,  $8k + 3$  ( $k = 0, 1, 2, 3$ ),  $P_{s2}$  is assigned  $8k + 4$ ,  $8k + 5$ ;  $P_{s3}$  is assigned segments  $8K + 6$ ;  $P_{s4}$  is assigned segments  $8k + 7$ , the start time is  $5t$ . however, if assignment(b) is used, the buffering delay will be reduced to  $4t$ .

We adopt an optimized media data assignment algorithm  $OTS_{p2p}$ [3] to select the needed pieces, and then send request to the peers in the list get from the tracker, And select the peer to download the needed pieces.

## 4 The Multi-agents Design

Because the character of peer to peer, the more peers who attend the same file download, the more available file resource for all peers. But at the beginning of the file transport, every node requests the initial file pieces from the seed peer who owns and distributes the entire media file. So the bottleneck lies in the beginning seed node. During the file transmission, the failure of seed node will lead to a very significant risk of a particular piece no longer being available from any current downloaders. During the transmission of the real time streaming media file, if some piece of the media file piece is rare, and the rare file piece is needed by some peer at the same time, it must affect the quality of the playback. To alleviate the seed peer burden and balance the distribution of the media pieces and improve the quality of media file transmission, we adopt the agent servers mechanism.

Our agent servers act as special peers and take the policy of rarest resource first to store the parts of a media file. With rarest resource first approach, each agent server will collect the information of pieces distribution among the peer including the other agent servers, then it selects the pieces which has the highest rarity among the peers to download and store. If a new file is issued among the overlay network, every piece of the file is rare, so every agent server will download pieces from the seed peer. Because the process of every agent server can be not synchronous, every agent server will download different pieces of the new mostly. To the file existed in the network, the agent server will store the rare pieces of the file to ensure the file integrity and the resource abundance.

The media piece replacement algorithm of Peer to Peer system is different from the general VOD(video on demand) system. In Peer to Peer system, if a media file becomes popular, there will be more downloaders to download the file, and the resource will become abundance, and the peers will get the media stream smoothly. The problem we will solve is to ensure the system have rich resource to the best effort. Our replacement algorithm is to make a binary decision based on the replacement parameter  $Q$ . The parameter  $Q$  of a media piece is based on the media exiguity extent and its popularity. When the number of a piece is equal or more than 3, the piece's value of the  $Q$  is set 1; when the pieces number



is less than 3 and the number of downloaders in some certain time is less than  $M$ , the piece's value of  $Q$  is set 0; if the disk of a proxy server is full then the proxy will random discard the pieces which pieces number is less than 3 and downloaders is more than  $M$ , those pieces' value of  $Q$  is 0 or 1. i.e.:

$$y = \begin{cases} 1 & \text{if } \textit{piecesnumber} > 3 \\ 0 & \text{if } \textit{piecesnumber} \leq 3 \text{ and the of } \textit{downloadernumber} < M \\ 0 \text{ or } 1 & \text{proxy disk is full and } \textit{piecesnumber} < 3 \text{ and } \textit{downloadernumber} > M \end{cases}$$

In practice, the threshold of *piecesnumber* and can be increased or decreased, and the length of time  $M$  can be adjusted. In this system, the threshold of *piecesnumber* is 3, and  $M$  is 24 hours.

## 5 Result and Analysis

We simulate the MPSS system model to validate the MPSS system. In first experiment, we contrast the system with the pure BitTorrent file sharing system on the rate of bandwidth utilization; then we verify the validity piece selection policy in the decrease media transport latency time; at last, we contrast the result that the system with media proxy and non-media proxy.

In this study, we used three test Networks, each representing a different degree of heterogeneity in the uplink and downlink capacities. The per-node uplink and downlink bandwidths of each group in each of the three networks are shown in Table 1.

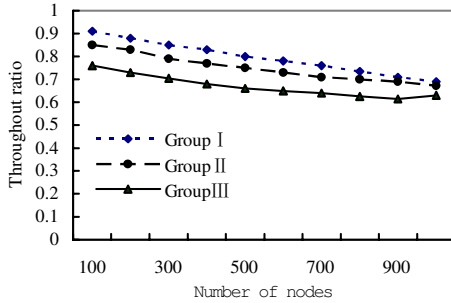
**Table 1.** Per-node upload and download bandwidth for the three different Networks

Group	Uplink Bandwidth	Downlink bandwidth	Percentage of nodes
1.1	384Kbps	1500Kbps	100%
2.1	128Kbps	768Kbps	50%
2.2	640Kbps	2232Kbps	50%
3.1	64Kbps	512Kbps	50%
3.2	384Kbps	1500Kbps	40%
3.3	1984Kbps	6440Kbps	10%

In the simulation, the uplink bandwidth of seed is set  $2000Kbps$ ; the number of proxy server is varies from 4 to 12, and the bandwidth of the proxy server  $100M$ , and the disk space is  $120G$ . The size of sharing file is  $256MB$ ; the size of each file piece is initialized  $256KB$ ; the size of each block is initialized  $16KB$ ; the number of receiver nodes is 400; the number of neighbors each node has is 40; and the maximal number of concurrent upload connection per node is 5. We contrast overall efficiency between the general BitTorrent system and the P2P-SIP system. We define the overall throughput ratio between BitTorrent and P2P-SIP as the evaluation parameter(formula(1)).

$$T_{atio} = \frac{Throughput_{BT}}{throughput_{p2p-sip}}$$

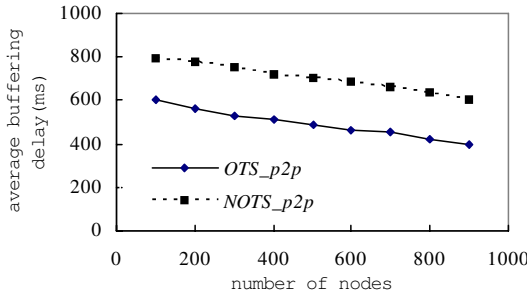
Figure 2 shows the overall throughput ratio between BitTorrent and MPSS with the number of node changing in three different Networks. From the figure, we can draw a conclusion that the overall throughput of the MPSS is obviously higher than that of BitTorrent. The higher bandwidth utilization efficiency of MPSS is benefit from the peer selection policy. Because the peer in MPSS chooses the peer with more bandwidth to upload media data, the overall throughput of the system is improved.



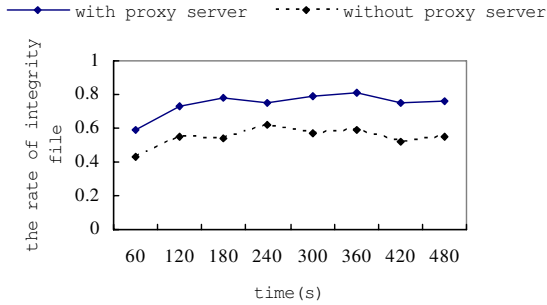
**Fig. 2.** Overall throughput ratio change between BitTorrent and MPSS with the number of node changing in three different Networks

Figure 3 shows the influence brought by the different piece selection. It is obvious that the MPSS with  $OTS_{p2p}$  piece selection policy has less buffering delay time than that of  $non - OTS_{p2p}$ . The system with MPSS has average 200ms buffering delay less than the  $non - OTS_{p2p}$ . The reason is because the  $OTS_{p2p}$  algorithm can compute an optimal media assignment, which achieves the minimum buffering delay in the consequent peer-to-peer streaming session.

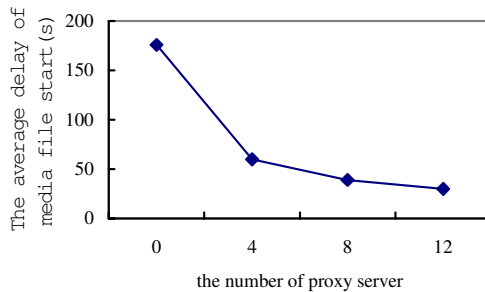
At last we test the change of media file play delay with the number of proxy server change. Figure 4 and figure 5 show importance of media proxy server. Without the media proxy server, the rate of file integrity is obviously lower than that of with proxy server. Some of the file cannot be gotten entirely at all, and the situation is improved by adopting the proxy server. Under the situation that



**Fig. 3.** Different influence brought by the piece selection



**Fig. 4.** The file integrity rate under the situation of with proxy server and without proxy server



**Fig. 5.** the change of start delay brought by media proxy server

the file can be fully gotten, the delay without proxy is obviously higher than that of with proxy. With the proxy server number increasing, the media buffering time is decreased.

## 6 Conclusion and Future Work

SIP is expected to be the communication standard in the future, and the peer to peer media stream system becomes more and more popular. In this paper, we combine the SIP family of IETF standards with Self Organizing properties of a Distributed Hash Table (DHT) P2P mechanism. To achieve the goal of playing while downloading, we apply the  $OTS_{p2p}$  media assignment algorithm and advanced peer selection mechanism. To get the fast capacity amplification of the entire peer-to-peer streaming and reduce the risk brought by the seed node is taken down, we adopt the media agent server mechanism. The simulation shows the MPSS system can meets the need for the distributed real time media communication.

The future work includes the encoded technique which will recover the original file. The Multiple Description Coding(MDC) which is used to encode the multimedia data to accommodate a set of heterogeneous client, will be the suitable choice. Another problem to be solved is the fairness, i.e. without decreasing

the overall throughout, how to let the peers who contribute the Networks most have the highest priority and discourage free riders.

## Acknowledgements

This work is supported by the HeBei province office of education Doctor Foundation under grant No.55470130-3.

## References

1. David A. Bryan, Bruce B. Lowekamp, and Cullen Jennings: SOSIMPLE: A Serverless, Standards-based, P2P SIP Communication System, Proceedings of the 2005 International Workshop on Advanced Architectures and Algorithms for Internet Delivery and Applications (AAA-IDEA 2005), (2005)
2. Rosenberg, J., Schulzrinne, H., Camarillo, G., Johnston, A., Peterson, J., Sparks, R., Handley, M. and E. Schooler: SIP: Session Initiation Protocol, RFC 3261, June (2002).
3. Dongyan Xu, Mohamed Hefeeda, Susanne E. Hambrusch, Bharat K. Bhargava: On Peer-to-Peer Media Streaming. ICDCS (2002) 363-371
4. Adar, E., Huberman, B.: Free Riding on Gnutella, First Monday, (2000), 5(10).
5. Cohen, B.: Incentives Build Robustness in BitTorrent. The 1st Workshop on Economics of P2P Systems, Berkeley, CA, (2003).
6. Vinay Pai, Kapil Kumar, Karthik Tamilmani, Vinay Sambamurthy, Alexander E. Mohr: Chainsaw: Eliminating Trees from Overlay Multicast. 4th International Workshop on Peer-to-Peer Systems. February (2005).
7. Ion Stoica, Robert Morris, David Liben-Nowell, David R. Karger, M. Frans Kaashoek, Frank Dabek, Hari Balakrishnan: Chord: A Scalable Peer-to-peer Lookup Protocol for Internet Applications. IEEE/ACM Transactions on Networking, Vol. 11, No. 1, (2003). 17-32.
8. Cao Le, Thanh Man, Go Hasegawa, Masayuki Murata: Passive and Active Network Measurement, 6th International Workshop, PAM 2005, Boston, MA, USA, 2005.

# Dynamic Context Aware System for Ubiquitous Computing Environment

Seungkeun Lee and Junghyun Lee

Department of Computer Science and Engineering  
Inha University, Incheon, Korea  
sglee@nlsun.inha.ac.kr, jhlee@inha.ac.kr

**Abstract.** Context aware system provides a process of context acquisition and reasoning to the context-aware service and enables developers to implement of these applications easily. Unfortunately, previous studies have not been able to support the user mobility, nor the service mobility efficiently. Both, however, should be supported in a ubiquitous computing environment. Therefore, this study proposes a dynamic context aware service model, which supports the management of context information and service mobility among service gateway. We also have designed a middleware based on this model. As the middleware is implemented on the OSGi framework, it can cause interoperability among devices such as computers, PDAs, home appliances and sensors. The proposed middleware has advantages of interoperation and integrating the heterogeneous devices by applying these standard interface technologies to this middleware.

**Keywords:** Context Aware, Ubiquitous Computing, Pervasive Computing.

## 1 Introduction

During the past 50 years, the concept of the human-computer interface has been evolved from multiple people sharing a single computer to a single individual using one computer, which has substantially changed the way humans use the computer system. The change has accelerated for the past decade to the point where a single person uses multiple computers or owns at least one computing device. Moreover, the ubiquitous computing technology has been studied in various aspects for environments where computers and devices are applied in everyday lives without being noticed[1, 2]. Moreover, the ubiquitous computing technology has been studied in various aspects for environments where computers and devices are applied in everyday lives without being noticed. Context is situational information that can be used to characterize the situation of an entity. An entity is a person, place, or object that is considered relevant to the interaction between a user and an application, including the user and application themselves. Context is a powerful, and longstanding, concept in human-computer interaction. Interaction with computation is by explicit acts of communication and the context is implicit. Context can be used to interpret explicit acts, making communication much more efficient. Communication can be not only effortless, but also naturally fit in with our ongoing activities[2, 3].

A context awareness service is usually based on the context awareness system or middleware. A context awareness system has a structure for generating the context information using the data acquired from the sensors and forwarding the data to the context awareness service. The context awareness services provide specific services to the user according to the context information received from the context awareness system. Therefore, the context awareness service must be able to clearly understand the significance of the context information conveyed from the context awareness system. There have been various studies on the techniques of context modeling, and the ontology modeling approach is receiving much attention, thanks to its capability to enable semantic exchange among systems[4].

However, conventional context modeling approaches based on ontology bear several problems. a context awareness service must share the context ontology with the context awareness system in the designing phase. In turn, when a context awareness service is dynamically added to or deleted from the system, the new context awareness service cannot share the context ontology with the system under the circumstances where other context awareness services are not affected. In addition, the problem of context uncertainty, which can arise in the process of deducing the data acquired from the sensor into the context information based on the dynamic modification of the context ontology, must be resolved.

Also, context awareness system is connected with a lot of external device such as sensors, actuators and appliance. So, this system need to designed based on common service gateway specification. OSGi is an industry plan regarding the standards for allowing sensors, embedded computing devices and electronic appliances to access the Internet, and it provides a Java-based open standard programming interface for enabling communication and control between service providers and devices within home and small business networks[5,6]. An OSGi-based system has a structure for distributing new services, and the structure consists of only the elements on the local network, giving it relatively closed characteristics. However, while service management and distribution can be dynamically executed within such single OSGi framework, there is insufficient support for applications with mobility among multiple frameworks. Therefore, there must be sufficient consideration for mobility of the users, devices and sensors among multiple OSGi frameworks in the expanded smart space, calling for research efforts in services supporting such mobility. Since the user, devices and sensors have mobility in a smart space, the services need to be mobile with their execution statuses stored. Also, mobility management must be efficiently supported for such service elements in the expanded smart space where multi-dimensional OSGi frameworks are located[1].

For example, let's say a user is heading toward home while listening to an mp3 music files on the PDA. Upon arrival at home, if the user wishes to listen to the same music on the PC, there is a cumbersome task of having to select the music play list from the PC directory. Even if there is an identical music play list in the PC, which eliminates the need for the user to select each individual song, there would not be the satisfaction of continuously listening to the music that had been playing on the PDA. However, if the mp3 player can be moved from the PDA to the PC, the music play list and song information can be maintained, allowing the user to appreciate the music without interruption[7][8][9].

This study deals with designing an context aware system based on OSGi framework that supports mobility and duplication with status information in the distribution environment. This system designed context manager and service mobility manager. Context manager is support a generation of context information from sensors which are connected by OSGi framework. Service mobility manger supports mobility of the bundles within multiple OSGi system environments. Therefore, it can support mobility of various elements such as services for specific components or users as well as device drivers. In order to do so, the OSGi framework's open source Knopflerfish 1.3.3 is expanded and a mobile agent management system is designed to support the bundles' mobile lifecycle[10].

## 2 Relative Works

This chapter addresses the context aware application and an overview of OSGi framework.

### 2.1 Context Aware Application

Conventional interactive computing provides a mechanism where the user directly inputs information into the computer. Consequently, the computer is unable to use any type of information that is generated from the human-computer interaction. By enhancing the level of understanding of the context of the computing environment, the human-computer interaction will bring better communication as well as more useful services to the users. In order to effectively utilize the context, we must first understand what the context is and how it can be used. Based on the understanding of the context, the application developer can determine which context is to be used for his development process. Understanding of how the context will be deployed allows the developer to identify the context awareness behavior that will be supported by his application. Schilit and Theimer first used the term "context awareness" in their study, which describes context as an identity that characterizes the location, individual or object, as well as environmental changes that include humans and objects[7]. Dey defines context as the emotional state, awareness, location, direction, date, time, people and objects that are associated with the environment surrounding the users[6].

### 2.2 OSGi

OSGi is a non-profit organization that defines standard specifications for delivering, allocating and managing services in the network environment. In its initial stage, OSGi was focused on the home service gateway, but it has recently expanded from a specific network environment to the ubiquitous environment. In turn, the objective of OSGi has become implementation of the service gateway for diverse embedded devices and their users[10].

The OSGi framework signifies the execution environment for such services and includes the minimum component model, management services for the components and the service registry. A service is an object registered in a framework used by other applications. For some services, functionality is defined by the interfaces they implement, allowing different applications to implement identical "service" types.

The OSGi framework installs an OSGi component called a “bundle” and supports a programming model for service registration and execution. The OSGi framework provides the mechanism for managing the bundle lifecycle. If a bundle is installed and executed in a framework, it can provide its services. Moreover, it can find and use other services available on the framework and can be grouped with other bundle services through the framework’s service registry. When registering a service in the OSGi framework’s service registry, a bundle stores the service attributes in the form of the attribute-value pair. Such service attributes are used so that they can be distinguished among multiple service providers.

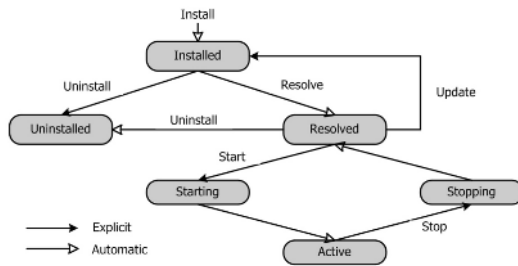


Fig. 1. The lifecycle of the bundle

### 3 Context Aware Service Middleware

Service management system consists of service mobility manage and service discovery manager. Figure 2 describes service management system.

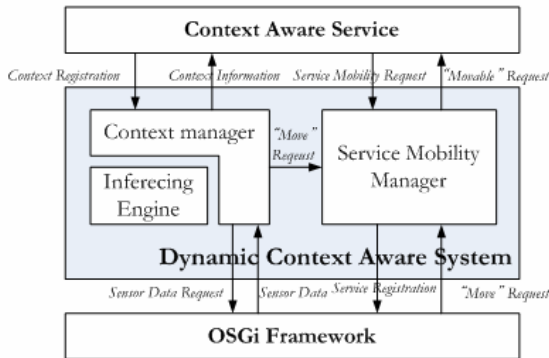


Fig. 2. The structure of the Service Mobility Manager

The overall structure consists of service mobility manager for managing mobility, service discovery manager for service matching and composition based on service semantic and ontology inference engine. If service is moved new OSGi framework,



service description is registered in service discovery manager. This service description is used in service matching and generation of service composition plan.

### 3.1 Service Mobility Manager

Service Mobility Manager consists of Mobility Interface, Service Serializer, SOAP Manager. Fig. 3. describes Service Mobility Manager. The overall structure consists of the mobile interface for managing mobility, elements for processing serialization/deserialization and elements for SOAP message transmission/reception. When the Mobility Interface receives a mobility request from a bundle service, it manages the service bundle’s lifecycle. The status information prior to mobility is marshalled into XML by the ServiceSerializer, and the SOAPClient delivers the SOAP message to the destination. The class file is installed through the destination BundleInstaller, and the ServiceDeserializer resumes the service by unmarshalling the SOAP message into an object.

#### The Extended Lifecycle for Service Bundle

In this paper, we extend the status of bundle in OSGi which compose of ‘Resolved’, ‘Starting’, ‘Active’ and ‘Stopping’, adding ‘DEAD’, ‘Movable’, ‘Moved’. ‘Dead’ status is different from ‘Uninstalled’ which means that bundles are omitted automatically. ‘DEAD’ is used for checking information of service before it is received move request. The status of bundle is changed ‘Movable’ to ‘Moved’ for the service which is received move request. ‘Move’ status is used for the process which OSGi framework sends service to other OSGi framework. If OSGi framework would finish sending of service to other OSGi framework, service status is changed to ‘Uninstalled’, and this bundle is removed.

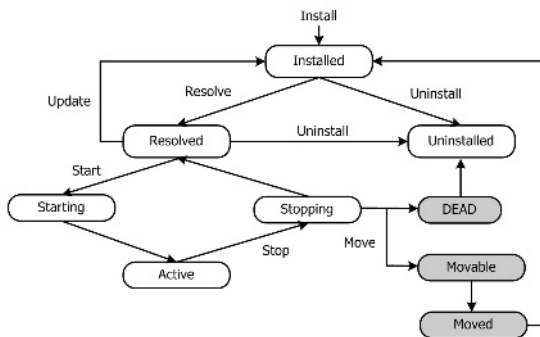


Fig. 3. The extended lifecycle for service bundle

#### Mobility of Service

Upon receipt of a mobility request, the service is switched to the mobility request state and execution suspension is requested. The service receiving the request returns after completing the action currently in process. If converted to the mobile state, the status information is serialized into the XML format using the Serializer, and service

mobility is requested to the SOAPClient. The SOAPClient verifies the destination and generates an SOAP message to call SOAPService to the destination. If mobility is successful, the service currently being executed is deleted from the registry.

At the destination, the SOAPService waiting for the SOAP message receives the URL information regarding the class location as well as the serialized data and delivers them to the MobileBundleManager. Prior to deserializing the received object, the MobileBundleManager installs the bundle from a remote location through the BundleInstaller. Upon successful installation, the object is deserialized and restored to the state prior to mobility. Finally, the service is converted to the RUNNABLE state and registered at the service registry. Algorithm 1 describes the process of service transmission between OSGi frameworks.

**Algorithm 1.** Sending ServiceObject with ServiceID

```

Input
ServiceID : ServiceID registered in Service
RegistryVariables
ServiceRef : Service Reference
ServiceDes : Service Description
ServiceStatus : Service Status Information
SOAPMessage : SOAPMessage used in Service Transmission
Begin Algorithm
  ServiceRef =
  ServiceManager.ServiceFinder.GetServiceRef(ServiceID)
  ServiceDes =
  ServiceManager.ServiceFinder.GetServiceDes(ServiceID)
  res = ServiceRef.beforeMoving()
  If (res is true)
  Begin If
    ServiceStatus = ServiceSerializer.serializer(res)
    SOAPMessage = SOAPService.makeSOAPMessage(URL,
      ServiceStatus, ServiceDes)
    sendMessage(TargetURL, SOAPMessage)
  End If
End Algorithm

```

When moving an object, it was sent to the SOAP Body element was a parameter by serializing it into an XML format as below. Castor and Apache Axis were used for serialization and SOAP transmission, respectively. Other resource and class files were not sent as attached files to the SOAP message. Rather, a mobile agent bundle was installed using the bundle allocation function at the remote location. The procedure and configuration for creating mobile bundles using the framework implemented in this paper are as follows. First, in order to implement a user-defined agent class, an XML schema must be created afterwards for object serialization using Castor in a format where the agent class defined in the agent bundle is inherited. The Manifest file of the user-defined agent designated the Bundle-Activator as the MobileBundleActivator, for which the ema.core.activator package in the agent bundle was imported. The MobileBundleActivator class reads the Agent-Class and Agent-

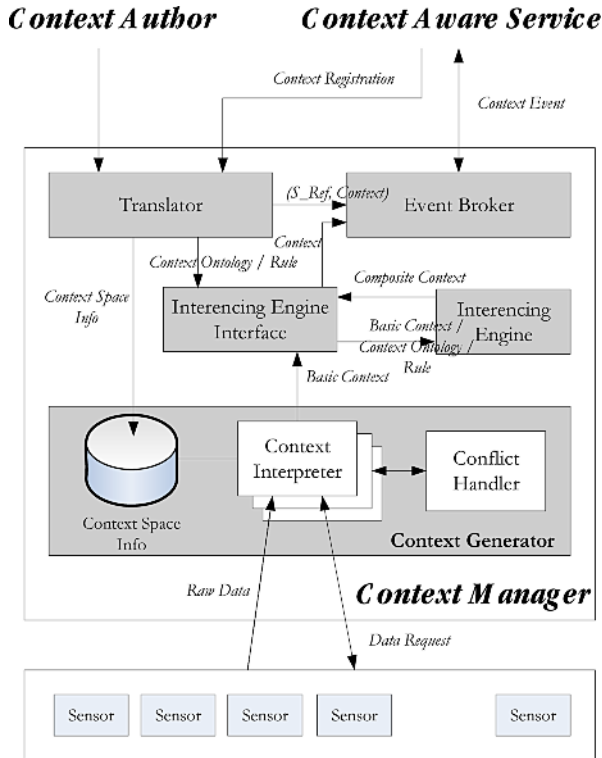
Name header values indicated in the Manifest using the bundle objective and registers them in EAR. In order to register a user agent as a service from the AgentManager bundle, the agent class registered in EAR must be dynamically loaded.

### 3.2 Context Manager

The context awareness middleware deduces the context information using the data received from the sensors and delivers the information to the desired context awareness service. In the development process of the context awareness service, the context awareness middleware collects, manages, combines the context information and forwards it to the context awareness services.

In order to deduce the context creator for creating the basic context and the basic context into complex contexts, the context awareness middleware is constructed into the ontology deduction engine, the deduction engine interface for interaction, the context awareness service, the event broker for delivering events in the supply/register format, and the translator for analyzing the commands received from the context author and the context awareness service. The author defines the context ontology required for the domain using the common context ontology defined in Chapter 3. The defined ontology is constructed with OWL files, which are sent to the translator. The translator provides a method for registering the context information types that the context awareness service needs. The context ontology defined by the context author and the rules for deducing the complex context are delivered to the context deduction engine through the deduction engine interface, and the context space information is stored in the database of the basic context generator. The basic context generator creates an appropriate basic context using the data collected from the sensors as well as the context space information stored in the database and the context conflict processor. The basic context information is delivered to the event deliverer through the deduction engine interface, converted into a fact for complex context deduction, and forwarded to the deduction engine. The basic context delivered to the deduction engine goes through the consistency test and the complex context deduction process using the rules and ontology defined by the context author. The generated complex context is again sent to the event deliverer through the deduction engine interface and finally to the context awareness service that requires the corresponding context information. Fig 3 is the system configuration of the overall context manager.

The context awareness middleware was designed as a bundle on the OSGi framework foundation. Therefore, the middleware inherits the BundleActivator interface of the OSGi framework and uses the Jena.jar file for ontology deduction. The translator receives the domain context information and rules from the ontology author, and the readFile() method that allows reception of individual context information in a file format from the context awareness service, internally parses the OWL file, which is delivered to the ontology deduction interface and the translator. The basic context translator compares the information acquired from the sensors and the context space information to generate the basic context. If there is a conflict among the context spaces, an appropriate context space is sought using resolveConflict() of the conflict processor.



**Fig. 4.** A Overview of Context Awareness Middleware

The context ontology information input by the middleware manager in the OWL file format and the basic context information generated by the context generator are transferred to Jena through the deduction engine interface. The deduction engine interface induces the complex context deduction using Jena, which is the ontology deduction engine. Here, insertContext() and removeContext() are for adding and deleting the basic context created by the basic context generator to and from the deduction engine in the Fact form. The complex context deduction rules defined by the middleware manager are managed using insertRule() and removeRule(). [List 1] describes the deduction engine interface.

**List 1.** Inferencing Engine Interface

```

Interface InferencingEngineInterface {
    FactID insertContext(Fact fact);
    boolean removeContext(FactID factID);
    RuleID insertRule(Rule rule);
    boolean removeRule(RuleID ruleID);
    boolean insertOntology(File ontologyFile);
    boolean removeOntology(File ontologyFile);
}
    
```

The deduction engine interface is constructed into six methods for adding and removing facts, rules and ontology. Each element receives an ID when being added to the deduction engine, and the ID is used to manage the corresponding element.

The basic context generator receives the data from the sensors and compares them with the context space information to deduce the appropriate basic context. The context information is given a context ID to be managed by the middleware, converted into a Fact through the deduction engine interface, and retransmitted to the deduction engine. The deduction engine checks consistency of the context information and generates complex context through ontology deduction and user rule-based deduction using the facts and the rules. The generated context information is delivered to the event broker, which forwards the information to the context awareness service that has registered the context information.

## 4 Experiment

We implement a smart homenetwork system based on dynamic context aware system proposed in this paper. The smart homenetwork has two small context aware service. One is the light control service which automatically turns on the light nearest to the user if the lighting is too dim. The light control service uses the information acquired by the illumination intensity sensor to recognize the brightness of each location. In order to assess the level of illumination in the house, light sensors are installed in Room1, Room2 and LivingRoom. The sensor in each location detects the light intensity and expresses it in 10 bits, creating a value between 0 and 1023. The context is deduced to be “dim” if the value is below 512, and “bright” otherwise. The light control service defines the independent contexts of the “user location dim” and the “light must be turned on” as in List 2.

**List 2.** The Rule for Context Inference

```
(?LightSensor locateIn ?place), (?LightSensorValue bigger 512) -> (?place lightLevel "Bright")
```

```
(?LightSensor locateIn ?place), (?LightSensorValue smaller 512) -> (?place lightLevel "Dim")
```

```
(?user locatedIn ?place), (?place lightLevel "Dim") -> (?place needed "lighting")
```

Second is Mobile MPlae which presents music service according to user’s location. During SOAP transmission from Knopflerfish using the Apache Axis, a minor bug was found in Java 1.5 or later, so the version 1.4.2 was used. In order to test the service management system proposed in this paper, an MPlayer bundle was developed for playing MP3 music files. JVM and OSGi service management system bundles were installed in a PDA and PC, respectively as an experiment environment. When a user listening to music from the MPlayer installed in the PDA moves into the space where the PC is located, the MPlayer service was moved to the PC and the file was resumed from the point where it had been playing from the PDA, providing a continuous MPlayer service to the user.

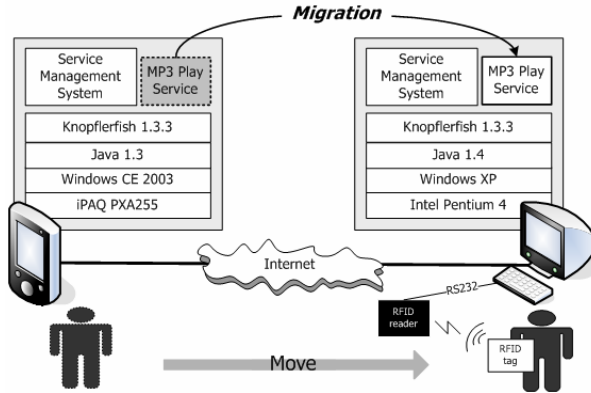


Fig. 5. The mobility of the MPlayer bundle

MPlayer in use from the PDA prior to service mobility. The state data serialized during mobility is the offset information regarding the music play list and the music file currently being played. The MPlayer does not have a GUI, and it is a bundle that plays the mp3 play list through a simple configuration file. Finally, The MPlayer bundle is automatically downloaded and installed using the bundle's class loading function, the service is initialized with the music play list offset data, and the music is played. And a description of this service is moved and registered in service registry. Implementation of the prototype displayed that the OSGi-based service management system proposed by this paper can operate as intended without much problem.

## 5 Conclusion

In this paper, we designed a dynamic context aware system which enables service and users to move among service gateways. This system is composed by context manager and service mobility manager. Context manager takes charge a generation of ontology based context information using sensor data. Service mobility manager is inevitable in order to provide object mobility among OSGi frameworks. This system is useful for the supporting of user and service mobility. This paper proposed a bundle in the form of a mobile service that can be autonomously executed in the OSGi framework, for which a mobile service lifecycle and a service mobility manager were designed and implemented for managing mobility. The designed context aware system was implemented in a bundle format to operate in the OSGi framework, and it also allowed dynamic management of autonomous services to provide mobility in a more efficient manner. In order to provide intelligent services in the future, there should be research efforts regarding OSGi-based situation recognition frameworks using the mobile service technology as well as security considerations for the mobile agent.

## Acknowledgement

This research was supported by the MIC(Ministry of Information and Communication), Korea, under the ITRC(Information Technology Research Center) support program supervised by the IITA(Institute of Information Technology Assessment) (IITA-2005-C1090-0502-0031).

## References

1. S. Lee et al, "Service Mobility Manager for OSGi Framework," Computational Science and Its Application 2006, LNCS3983, Springer, 2006.
2. N. Q. Hung, S. Y. Lee, and L. X. Hung, "A Middleware Framework for Context Acquisition in Ubiquitous Computing Systems", Proceedings of the Second International Conference on Computer Applications, 2004.
3. A. Padovitz, S. W. Loke, and A. Zaslavsky, "Towards a Theory of Context Spaces," Proceedings of the Second IEEE Annual Conference on Pervasive Computing and Communications Workshops, 2004.
4. [Gu04] T. Gu, H. K. Pung, and D. Q. Zhang, "An Ontology-based Context Model in Intelligent Environments," Proceedings of Communication Networks and Distributed Systems Modeling and Simulation Conference, pp. 270-275, 2004.
5. Open Services Gateway Initiative. <http://www.osgi.org>
6. D. Marples and P. Kriens, "The Open Services Gateway Initiative: An Introductory Overview," IEEE Communications Magazine, Vol. 39, No. 12, pp.110-114, December 2001
7. C. Lee, D. Nordstedt, and S. Helal, "Enabling Smart Spaces with OSGi," IEEE Pervasive Computing, Vol. 2, Issue 3, pp.89-94, July\_Sept. 2003
8. K. Kang and J. Lee, "Implementation of Management Agents for an OSGi-based Residential Gateway," The 6th International Conference on Advanced Communication Technology, Vol. 2, pp.1103-1107, 2004
9. F. Yang, "Design and Implement of the Home Networking Service Agent Federation Using Open Service Gateway," International Conference on Integration of Knowledge Intensive Multi-Agent Systems, pp.628-633, Sept.\_Oct. 2003
10. Knopflerfish. <http://www.knopflerfish.org>

# Partial Group Session Key Agreement Scheme for Mobile Agents in e-Commerce Environment\*

Hyun-jin Cho, Gu Su Kim, and Young Ik Eom

School of Information and Communication Eng., Sungkyunkwan University,  
300 cheoncheon-dong, Jangan-gu, Suwon, Gyeonggi-do 440-746, Korea  
{hjcho, gusukim, yieom}@ece.skku.ac.kr

**Abstract.** In this paper, we consider an e-Commerce system that supports buying and selling goods or services over the Internet using Mobile Agent(MA) technologies. MAs are active, autonomous, and self-replicable software objects containing both computational logic and state information. When MA technologies are applied to e-Commerce environments, the MAs can search for a product on behalf of the buyer. The buyer constructs a group of MAs, and dispatches each agent in the group to the supplier sites. Each buyer agent compares price information of goods which are suggested by the suppliers. Eventually, all MAs exchange their price information cooperating to determine the most suitable item. In the selection process, if the price information which is exchanged among the MAs is modified by malicious entities, the user may not select intended goods. In this paper, we propose a group session key generation scheme called PGKA(Partial Group session Key Agreement) for secure communication among the MAs. The PGKA has no sponsors, controllers, or any other members charged with special duties. The main idea in PGKA is to distribute the computation of the group session key generation among the members, which makes the scheme suitable for fully distributed environments. In addition, the cost of the key reconstruction is very low in this scheme, because only the seed value is required to be transmitted for the reconstruction process.

## 1 Introduction

Mobile agents(MAs) are active and autonomous software objects containing both computational logic and state information. The MAs run on one host, migrate to another host with their state, and continue execution on that host. The advantages of using MAs include low network traffic usage, asynchronous parallel execution, simple implementation, simple deployment, high reliability, and so on [1].

MAs can be utilized to assist the automation of e-Commerce environments and to improve the trade performance among the buyers and suppliers. One of the primary issues of e-Commerce is the difficulty in integrating supply chains among the enterprises, which is carried out to reduce inventories and promote

---

\* This work was supported by National Center of Excellence in Ubiquitous Computing and Networking (CUCN), Korea.



transaction automation. Agent-based technology provides a solution to this issue [2-5].

The current e-marketplace architecture is usually based on the centralized transactional system. All commerce actions occur on the same server, including searching products, asking for quotations, and negotiating. However, because current network environments are basically open and distributed, the e-Commerce systems should be developed based on open and distributed system environments.

In distributed e-marketplace systems, the buyer can dispatch multiple agents to supplier sites and negotiate with supplier agents. The buyer constructs an agent group consisting of multiple MAs and dispatches each agent in the group to each supplier site. The MA arriving at the supplier site retrieves price information of goods through communication with the supplier agents. Finally, all MAs in the group exchange price information of the goods with each other and determine the most suitable item [2].

However, during MA communication, if private information is revealed and modified, the user may not select the intended items. Therefore, the communication among the MAs in the group should be protected from eavesdropping by malicious entities. In secure group communications, efficient group key management is one of the central issues. We propose a PGKA, a group session key generation scheme for secure communication, based on partially shared information of the MAs in the group. In a PGKA, each member generates a group session key using three key materials, among which, two key materials are shared with other members of the group and the other is private key material that is not shared in the group. Each member generates a group session key by appending its own key material to the two shared key materials.

The subsequent sections of the paper are organized as follows. In Section 2, the e-Commerce system based on MAs and the existing group session key generation scheme is described. Section 3 presents the proposed PGKA (Partial Group session Key Agreement) scheme. Section 4 presents the safety analysis of the PGKA scheme and Section 5 compares the operation cost of the PGKA scheme with three of the existing schemes. Finally, Section 6 concludes with a summary.

## **2 e-Commerce Systems Based on MAs and Group Session Key Generation Scheme**

In this Section, we describe a brief overview of e-Commerce systems based on MAs and existing group session key generation schemes for secure group communication.

### **2.1 e-Commerce Systems Based on MAs**

There are several e-Commerce systems based on MAs [2-5]. With the scheme proposed in [2], the buyer could dispatch multiple agents to the supplier sites and negotiate with the supplier agents. All buyer agents exchange and share

the quotation of all supplier agents to reach an agreement. In MAgNET [3], a buyer maintains a list of potential suppliers along with their lists of products. A buyer interested in acquiring a product, creates an MA, provides it with an itinerary of the supplier sites, specifies the criteria for the acquisition of the product, and dispatches the MA to the potential suppliers. The MA visits each supplier site, searches product catalogs according to the buyer's criteria, and returns with the most suitable product to the buyer site. In [4], a two-layered multi-agent framework is proposed for brokerage between buyers and sellers. The brokerage is processed in two layers for efficient linking between buyers and sellers: the competition layer and the constraint satisfaction layer. Nomad [5] allows mobile agents to travel to the eAuctionHouse [6] site and actively participate in auctions on the user's behalf even when the user is disconnected from the network.

## 2.2 Group Session Key Generation Schemes

The group session key generation schemes for secure group communication can be classified into three main classes: centralized group key management scheme, decentralized group key management scheme, and distributed group key management scheme [7-9].

In centralized group key management scheme, a group controller controls the entire group and the group members rely on the group controller. However, this scheme has shortcomings. First, there is some overhead in managing the secret key, which is used to deliver the group session key to each member. Furthermore, the group may become too large to be managed by a single controller.

In decentralized group key management scheme, the management of a large group is divided into subgroup controllers. Each controller manages each subgroup, minimizing the problem of concentrating the work at a single location. In this approach, more entities are permitted to fail before the whole group is affected. However, each subgroup should manage a group session key in the same way as the centralized group key management scheme. This approach is not suitable for small group environments.

In distributed key management scheme, there is no explicit group controller. The group session key can also be generated in a contributory fashion, where all members contribute to the computation of the group key. The processing time and communication requirements of most contributory schemes increase proportionally with the number of members. In addition, the contributory protocols require each user to be aware of the group membership and ensure the protocols to be robust.

## 3 PGKA Scheme

In this Section, the PGKA scheme is described, which is a derivative of the distributed group session key generation scheme. The proposed scheme is based on the partial shared information of the group.

### 3.1 System Architecture

Our e-Commerce architecture is based on cooperative multi-agent negotiation [2]. Figure 1 presents the system architecture and the scenario of buying goods using the MAs.

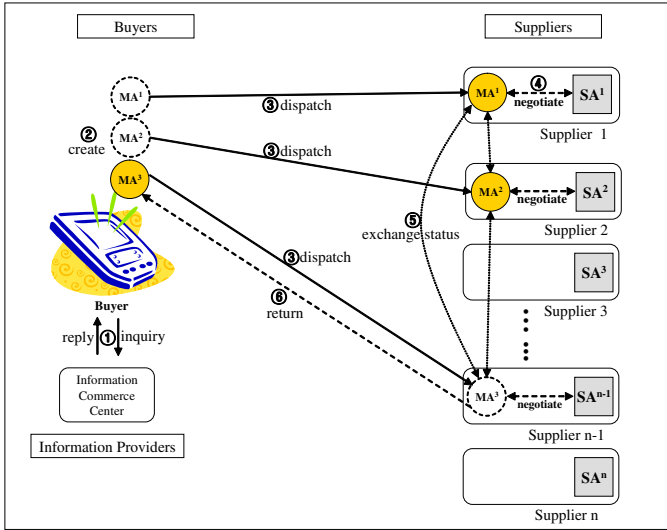


Fig. 1. e-Commerce architecture and the scenario of buying goods

It is assumed that the e-marketplace consists of buyers, supplier sites, and a commerce center providing the supplier information. All supplier sites have their own supplier agents. If a buyer requests a quotation of certain goods, an inquiry is transmitted to the supplier section of the commerce center and the commerce center responds with a list of supplier sites for the buyer. The buyer composes multiple buyer agents and dispatches each agent to each supplier site in the list. The buyer agents retrieve quotations of the products from the supplier agents at each supplier site and exchange this information with each other. The most suitable item is then determined.

In this scenario, the communication among the buyer agents should be protected from eavesdropping by malicious entities. For secure communication among the MAs in a group, the communication messages can be encrypted using the group session key.

### 3.2 Group Session Key Generation

As described previously, buyer agents are dispatched to the supplier sites and exchange the quotation information of the goods internally to decide the most suitable item. If this information is revealed and/or modified by malicious entities, the buyer may purchase the wrong items. In this subsection, we describe

a group session key generation scheme using a partial group session key that is shared among members, without a group controller.

When the buyer composes an MA group, the group ID and the random number generator(RNG) are inserted into each member. The group ID proves that each MA with same group ID is the member of the same group. Also, each member verifies integrity of group session key generation message using hash value of the group ID. The RNG generates partial key materials for group session key generation. In this paper, we assume that the RNG follows ANSI X9.17 PRNG(Pseudo Random Number Generator), which is a specification of cryptographically secure pseudorandom number generator[10].

The PGKA scheme uses three key materials for group session key generation:  $\alpha$ ,  $y$ , and  $x$ .  $\alpha$  is a common key material. All members in the mobile agent group share the same value of  $\alpha$ .  $y$  is a public key material, where each member creates its own  $y$  and exchanges it with the other members in the group.  $x$  is a private key material. In contrast with  $y$ ,  $x$  is not shared with the other members. Even if  $\alpha$  and  $y$  is disclosed to the malicious object, each member generates a group session key securely because only the owner knows the value of  $x$ .

In order to generate group session key, the user dispatches MAs to the supplier sites to obtain the information of goods. After the MAs are dispatched to each supplier site, if an MA, denoted by  $MA^k$ , wants to communicate with the other members, the  $MA^k$  chooses a  $seed\_value_k$  and  $nonce_k$  to generate a 128-bit key material  $\alpha$ . Both the  $seed\_value_k$  and  $nonce_k$  are 64-bit value and they are input value to the RNG for  $\alpha$ . The  $MA^k$  also generates an  $x_k$  using the new  $seed\_value$  and  $nonce$ . Thereafter, the  $MA^k$  computes  $y_k$  with  $x_k$  and  $\alpha$  using Strong Associative One-Way Function(Strong-AOWF) [11] and encrypts  $y_k$  with  $\alpha$ . The Strong-AOWF is associative and commutative. In this paper we use  $\circ$  as a Strong-AOWF. Then,  $y_k$  can be computed as follows:

$$y_k = x_k \circ \alpha = \alpha \circ x_k.$$

Finally, the  $MA^k$  generates a *GeneratePartialKey* message and transmits it to the other members. The *GeneratePartialKey* message is structured as follows:

$$\langle packet\_type, seed\_value_k, E_\alpha(y_k), nonce_k, \\ H[G_{id}|seed\_value_k|E_\alpha(y_k)|nonce_k] \rangle.$$

Table 1 shows description of each parameter in the *GeneratePartialKey* message. When the *GeneratePartialKey* message is transmitted to the other members, they are requested to generate a group session key, and generate  $\alpha$  using  $seed\_value_k$  and  $nonce_k$  included in the *GeneratePartialKey* message. Then, each member generates its own random number  $x_i$ , which represents a private key material. Also, each member generates a public key material  $y_i$  with  $x_i$  and  $\alpha$  using Strong-AOWF, encrypts  $y_i$  with  $\alpha$ , and responds to the other members with the *ResponsePartialKey* message. The *ResponsePartialKey* message is structured as follows:

$$\langle packet\_type, E_\alpha(y_i), nonce_i, H[G_{id}|E_\alpha(y_i)|nonce_i|nonce_k] \rangle,$$

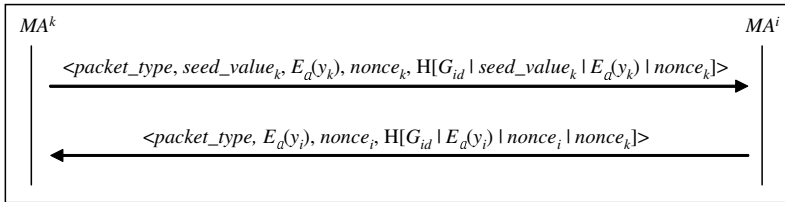
where  $nonce_i$  and  $y_i$  is nonce and public key material of each MA, respectively.

**Table 1.** Description of each parameter in the *GeneratePartialKey* message

Parameters	Description
$packet\_type$	Message identifier
$seed\_value_k$	64-bit input value for creation of $\alpha$
$E_d(y_k)$	Encryption of $y_k$ by $\alpha$
$nonce_k$	64-bit input value for creation of $\alpha$ and prevention of replay attacks
$H[G_{id}   seed\_value_k   E_d(y_k)   nonce_k]$	Hash function for prevention of masquerade attacks

The members receiving the *ResponsePartialKey* message decrypts this message with  $\alpha$ . Figure 2 presents the process of message exchange for the public group session key material.

Each member generates a group session key, denoted by  $G_k$ , using its own private key material  $x_i$  and public key material  $y_i$ . Figure 3 presents the process of group key generation. Because the Strong-AOWF is associative and commutative, the group session keys generated by the members in a group are identical. Hereafter, all communication messages in the group are encrypted by  $G_k$ .



**Fig. 2.** The process of message exchange for public partial group session key material

$G_k = x_1 \circ y_2 \circ y_3 \circ \dots \circ y_{i-2} \circ y_{i-1} \circ \dots \circ y_n$ $= x_1 \circ (x_2 \circ a) \circ (x_3 \circ a) \circ \dots \circ (x_{i-2} \circ a) \circ (x_{i-1} \circ a) \circ \dots \circ (x_n \circ a)$
<p><math>\circ</math> : Strong Associative One-Way Function (Strong-AOWF)</p>

**Fig. 3.** The process of group session key generation

### 3.3 Refreshment of the Group Session Key

The group session key is reconstructed when any of the followings occur:

- The lifetime of the key expires
- A new member joins the group
- An existing member withdraws its group membership

The purpose of the group session key refreshment is to protect the secure multicast communications from group key exposure due to long-time use of group key and from key exposure by withdrawn members [12]. Moreover, the PGKA regenerates the group session key when a new member joins the group. When the group session key lifetime expires, each member generates a new group session key with the previous public key material and common key material. The expiration time of the key is fixed when the group is organized. The new public key material is generated as follows:

$$y_n^{new} = y_n^{old} \circ \alpha^{new},$$

where  $y_n^{old}$  and  $y_n^{new}$  represent the old and new public key material, respectively. In the above formula,  $\alpha^{new}$  represents a new common key material generated with the new *seed\_value* and *nonce*. The new group session key is generated as follows:

$$G_k^{new} = y_1^{new} \circ y_2^{new} \circ \dots \circ y_{i-2}^{new} \circ y_{i-1}^{new} \circ \dots \circ y_n^{new}.$$

If a buyer is aware of a new supplier site and desires obtaining a quotation of the goods located on that supplier site, the buyer creates a new buyer agent with the same group ID and dispatches it to the new supplier site. The new group member, denoted by  $MA^{new}$ , contacts the nearest member, denoted by  $MA^y$ . The  $MA^y$  encrypts the previous public key material with  $\alpha^{new}$  and generates a *PrevPartialKey* message, which is sent to the  $MA^{new}$ . The *PrevPartialKey* message is structured as follows:

$$\langle packet\_type, seed\_value, E_{\alpha^{new}}(y_n^{old}), nonce_y, \\ H[G_{id}|seed\_value|E_{\alpha^{new}}(y_n^{old})|nonce_y] \rangle.$$

The  $MA^{new}$  generates  $\alpha^{new}$ , decrypts  $E_{\alpha^{new}}(y_n^{old})$ , and generates a new group session key. The  $MA^y$  inserts the ID of the new member into the *ReconstructKey*<sup>new</sup> message and multicasts it to existing group members. The *ReconstructKey*<sup>new</sup> message that is multicasted by the  $MA^y$  is structured as follows:

$$\langle packet\_type, seed\_value, MA^{new}, nonce_y, \\ H_{G_k}[G_{id}|seed\_value|MA^{new}|nonce_y] \rangle,$$

where the hash function in the *ReconstructKey*<sup>new</sup> message is keyed hash function with group session key. The other members also generate  $\alpha$  with the *seed\_value* and *nonce*, and generate a new group session key.

If an existing member withdraws from a group membership, the member, denoted by  $MA^{old}$ , notifies the withdrawal request to the nearest member, denoted

by  $MA^z$ . At this time, the  $MA^z$  generates a  $ReconstructKey^{old}$  message, which includes the ID of the leaving member, and multicasts it to the other members. The  $ReconstructKey^{old}$  message that is multicasted by the  $MA^z$  is structured as follows:

$$\langle packet\_type, seed\_value, MA^{old}, nonce_z, H_{G_k}[G_{id} | seed\_value | MA^{old} | nonce_z] \rangle.$$

Also, the keyed hash functions is used in the  $ReconstructKey^{old}$  message with group session key. Other group members generate  $\alpha$  with the  $seed\_value$  and nonce, and generate a new group session key.

## 4 Safety Analysis of the PGKA Scheme

In this Section, the safety of the PGKA scheme is analyzed. Table 2 and Table 3 describe the information and messages related to group session key generation.

### 4.1 The Attack on the Message for Group Session Key Generation

Malicious entities may impersonate a group member and transmit the  $GeneratePartialKey$  message to the other members. However, malicious entities do not

**Table 2.** Information for group session key generation

Information	Description
$\alpha$	A common key material which is generated by each member with $seed\_value$ and $nonce$
$y_i$	A public key material which is exchanged with the other members in the group
$x_i$	A private key material which is not shared with the other members in the group

**Table 3.** Messages for group session key generation

Message	Formation	Main information
$GeneratePartialKey$	$\langle packet\_type, seed\_value_p, E_d(y_k), nonce_k, H[G_{id}   seed\_value_k   E_d(y_k)   nonce_k] \rangle$	The message notifying generation of a group session key
$ResponsePartialKey$	$\langle packet\_type, E_d(y_i), nonce_p, H[G_{id}   E_d(y_i)   nonce_i   nonce_k] \rangle$	The message including the public key material for group session key generation
$PrevPartialKey$	$\langle packet\_type, seed\_value, E_{d^{new}}(y_n^{old}), nonce_y, H[G_{id}   seed\_value   E_{d^{new}}(y_n^{old})   nonce_y] \rangle$	Key reconstruction message for new member
$ReconstructKey^{new}$	$\langle packet\_type, seed\_value, MA^{new}, nonce_y, H_{G_k}[G_{id}   seed\_value   MA^{new}   nonce_y] \rangle$	Key reconstruction message multicasted when a new member joins
$ReconstructKey^{old}$	$\langle packet\_type, seed\_value, MA^{old}, nonce_z, H_{G_k}[G_{id}   seed\_value   MA^{old}   nonce_z] \rangle$	Key reconstruction message multicasted when an existing member leaves

have a group ID. The hash value of group ID is inserted into the group session key messages for prevention of masquerade attack. Therefore, malicious entities can not fabricate fake messages for group session key generation. Also, malicious entities can not generate the common key material  $\alpha$  because they do not have the RNG for generating  $\alpha$ . If a malicious entity obtain a different  $\alpha$  with the other members, it can not decrypt  $E_\alpha(y_i)$  in the group key generation message.

#### 4.2 The Attack Against the Group Session Key Update

New group session key is generated with  $y_n^{old}$  and  $\alpha^{new}$ . Therefore, malicious entities should obtain these information to generate the same group session key. But, the  $y_n^{old}$  and  $\alpha^{new}$  is transmitted only to the new member when the group membership is changed. Eventually, the malicious entities can just obtain the  $y_n^{old}$  through the *PreviousPartialKey* message. However, the  $y_n^{old}$  in the *PreviousPartialKey* message is encrypted with  $\alpha^{new}$ . According to the previous section, the malicious entities do not have the RNG for generation of  $\alpha^{new}$ . Also, the malicious entities can not predict the group session key reconstruction time because it is reconstructed periodically or when the group membership is changed.

### 5 Performance Comparison of the PGKA Scheme with the Existing Schemes

In this Section, the costs of the four group session key generation schemes are analyzed as presented in Table 4~6.

Table 4 presents the comparison of the performance of PGKA scheme with the other existing schemes [7] such as Burmester-Desmedt (BD) [13], Group Diffie-Hellman (G-DH) [14], and Distributed Logical Key Hierarchy (D-LKH) [15].

Among the group session key generation schemes, BD, G-DH, and PGKA have no group leader. Those schemes have several advantages over the group session key generation schemes that needs a group leader. In the group session key generation schemes without group leader, all members are treated equally and, if one or more members fail to complete the setup, it will not affect the whole group. In those schemes with a group leader, the group leader failure is

**Table 4.** Communication and computation costs for group session key Generation

Scheme	Setup		Rounds	No. of messages		Key reconstruction time
	Leader	Other		Multicast	Unicast	
BD	-	$(n-1)E_x$	3	$2n$	0	Membership changed
G-DH	-	$(i+1)E_x$	$n$	$n$	$n-1$	Membership changed
D-LKH	$\log_2 n E_x$	$\log_2 n D$	3	1	$n$	Membership changed
PGKA	-	$E+(n-1)D+nX$	3	$n$	0	Membership changed or the lifetime of the key expires



**Table 5.** Notation for Table 4

Notation	Description
$n$	No. of group members
$i$	Index of member
$E$	Encryption
$D$	Decryption
$E_x$	Exponentiation
$X$	Strong-AOWF operation

fatal for the entire group. In the D-LKH, the members are organized into a logical tree hierarchy, which each subtree has a group leader.

BD, D-LKH, and PGKA have a fixed number of rounds. That is, the number of interactions for group session key generation among the members are independent of the number of members in the group. In PGKA, the MA desiring communication with the other members, generates a *GeneratePartialKey* message and multicasts it to the other members. The other members that receive the *GeneratePartialKey* message multicast the *ResponsePartialKey* message to the other members. Finally, the members receiving the *ResponsePartialKey* generate the group session key. Therefore, generation of the group session key is completed in 3-rounds.

During the group session key generating process,  $n$  times multicast has occurred where  $n$  is the number of members in the group. For generation of a group session key, each member employs the following operations: 1 encryption,  $(n - 1)$  decryptions, and  $n = 1 + (n - 1)$  Strong-AOWF operations. The number of messages required for generation of group session key relies on the existence of the group leader. If the group leader is in existence, unicast messages increase, while, in the other case, multicast messages increase.

**Table 6.** Communication costs for group membership change

Scheme	Membership change	Rounds	Multicast	Unicast
BD	Join	2	$2n+2$	0
	Leave	2	$2n-2$	0
G-DH	Join	4	2	$n+1$
	Leave	1	1	0
D-LKH	Join	2	2	1
	Leave	2	$\log_2 n$	$\log_2 n$
PGKA	Join	3	1	2
	Leave	2	1	1

Table 6 presents the communication costs generated when the group membership changes. In the table 6,  $n$  denotes the number of group members. When a new member is added in the group, G-DH has maximum number of rounds while BD and D-LKH have minimum number of rounds. BD uses  $n$  broadcast messages for each round. Therefore BD most frequently exchanges messages when a new member is added in the group. D-LKH is characterized by stable communication costs relative to the other schemes. D-LKH has group leaders, because the members of a group in the D-LKH are organized into logical tree hierarchy. PGKA reconstructs the key not only when the group membership changes, but also periodically. Therefore, the group session key is more secure than the other schemes. In addition, PGKA uses the previous public key material to generate a new group session key. Therefore, in the PGKA, the communication costs for generation of a new group session key are very low.

## 6 Conclusion

In this paper, a group session key generation scheme, called PGKA, is proposed. This scheme is based on a partial group session key for secure group communication. In the PGKA scheme, there is no group controller, creating and distributing group session keys. Each member generates a group session key using its own private key material and the other member's public key materials. In the PGKA scheme, the group session key is reconstructed periodically or when group membership changes. The new group session key is generated using the other member's public key materials generated from previous public key materials, with a new random number. Therefore, only the seed value is required to be transmitted for the reconstruction process.

## References

1. A. Aneiba and J. S. Rees, "Mobile Agent Technology and Mobility," *Proc. of the 5th Annual Postgraduate Symposium on the Convergence of Telecommunications, Networking and Broadcasting*, Jun. 2004.
2. F. C. Lin and C. N. Kuo, "Cooperative Multi-Agent Negotiation for Electronic Commerce Based on Mobile Agents," *Proc. of the IEEE Int'l Conf. on Systems, Man and Cybernetics*, Oct. 2002.
3. P. Dasgupta, N. Narasimhan, L. E. Moser, and P. M. Melliar-Smith, "MAGNET: Mobile Agents for Networked Electronic Trading," *Proc. of the IEEE Int'l Conf. on Transactions on Knowledge and Data Engineering*, Jul./Aug. 1999.
4. J. J. Jong and S. J. Geun, "Brokerage Between Buyer and Seller Agents using Constraint Satisfaction Problem Models," *Proc. of the Decision Support Systems, Vol. 28, Issue 4, Elsevier Science*, Jun. 2000.
5. T. Sandholm and Q. Huai, "Nomad: Mobile Agent System for an Internet-Based Auction House," *Int'l Journal of the IEEE Internet Computing*, Mar./Apr. 2000.
6. T. W. Sandholm, "eMediator: A Next-Generation Electronic Commerce Server," *Proc. of the Fourth Int'l Conf. Autonomous Agents(AGENTS)*, Jun. 2000.
7. S. Rafaeli and D. Hutchison, "A Survey of Key Management for Secure Group Communication," *ACM Computing Surveys*, Sep. 2003.

8. M. Moyer, J. Rao, and P. Rohatgi, "A Survey of Security Issues in Multicast Communications," *Int'l Journal of the IEEE Network*, Nov./Dec. 1999.
9. Y. Amir, Y. Kim, C. Nita-Rotaru, and G. Tsudik, "On the Performance of Group Key Agreement Protocols," *Proc. of the 22nd. Int'l Conf. on Distributed Computing Systems*, Jul. 2002.
10. W. Stallings, *Cryptography and Network Security; Principles and Practice*, Prentice Hall, 2003.
11. M. Rabi and A. T. Sherman, "Associative one-way functions: A New Paradigm for Secret-Key Agreement and Digital Signatures," Technical Report CS-TR-3183/UMIACS-TR-93-124, Dept. of Computer Science, Univ. of Maryland, College Park, Maryland, Nov. 1993.
12. G. Y. Lee, Y. Lee, "Efficient Rekey Interval for Minimum Cost on Secure Multicast System using Group Key," *Proc. of the IEEE Int'l Conf. on Global Telecommunications Conf. (GLOBECOM 02)*, Nov. 2002.
13. M. Burmester and Y. Desmedt, "A Secure and Efficient Conference Key Distribution System," *Advances in Cryptology: Proc. of the CRYPTO '94*, Lecture Notes in Computer Science 839, Springer-Verlag, Berlin, 1994.
14. M. Steiner, G. Tsudik, and M. Waidner, "Diffie-Hellman Key Distribution Extended to Group Communication," *Proc. of the 3rd ACM Conf. on Computer and Communications Security*, Mar. 1996.
15. O. Rodeh, K. P. Birman, and D. Dolev, "Optimized Group Rekey for Group Communication Systems," *Proc. of the Symp. on Network and Distributed Systems Security (NDSS '00)*, Feb. 2000.

# Optimal Agendas for Sequential English Auctions with Private and Common Values

Yu-mei Chai<sup>1</sup> and Zhong-feng Wang<sup>2</sup>

School of Information & Engineering,  
Zhengzhou University,  
450052 Zhengzhou, China

<sup>1</sup> ieymchai@zzu.edu.cn,

<sup>2</sup> iewzf@163.com

**Abstract.** This paper studies how to relax the restrictions on bidders in two sequential English auctions. Existing work has showed that it is optimal for the auctioneer to auction objects in the decreasing order of the dispersion of the order statistics of the surpluses. But they assume each bidder can win at most one object. However, in many cases, each bidder needs more than one object and the objects almost affect each other. Given this, the settings that have both common and private value elements by allowing each bidder winning more than one object are studied here. Firstly, the single object model is extended to two objects without the limitation that a bidder can win at most one object. Secondly, the equilibrium bidding strategies for each auction in a sequence are described, the selling prices and winner's total expected profits of different agendas are analyzed. Thirdly, the optimal agendas are determined. Finally, we make two experiments, one validates that it may be the optimal agenda either by increasing order or decreasing order of the dispersion of the order statistics of the surpluses, the other display the changing trend of the optimal agenda.

**Keywords:** Multi-agent Systems, sequential English auctions, optimal agendas.

## 1 Introduction

As a kind of market mechanism, auction can find a reasonable value for the objects which are difficult to confirm their selling price [1][2]. Early in 1961, Vickrey [3] groups the auction into English auction, Dutch auction, first-price auction and second-price auction. In 1981, Riley and Samuelson [4] present the expectation income equivalence theorem of the four basic auctions under the model SIPV. But in practice, the goods on sale are not always single. In such cases, two types of auctions can be used: combinatorial auctions [5][6][7] and sequential auctions [8][9]. The former are used when the objects for sale are available at the same time, while the latter (which is the main focus in this paper) are used when the objects become available at different points in time.

W. Elmaghraby [10] introduce and highlight the critical influence that ordering can have on the efficiency of an auction, Jean-Pierre Benoit *et al.* [11] proposed that if just considering the common value, it is always optimal to sell the more valuable

object first for heterogeneous objects with budget constrained bidders. Mireia Jofre-Bonet and Martin Pesendorfer[12] examine which auction rule achieves the low procurement cost. They show that the answer to this policy question depends on whether the items are complements or substitutes only thinking of private value, S.S.Fatima *et al.* [13] studied the optimum auction agenda, considering common value and private value at the same time. They built up a multi-objects-auction model on the basis of the model put forward by J.K.Goeree and T.Offerman[14]. Analysis of the model, they proposed that it is optimal for the auctioneer to auction objects in the decreasing order of the dispersion of the order statistics of the surpluses. However, S.S.Fatima *et al.* [13] [15] [16] assumed each bidder can be the winner with only one time. This deserves bargaining, because first J.K.Goeree and T.Offerman[14] testified that increased competition raises the auctioneer's expected revenue on single-object auction, second the relation of the objects that the bidders are assured to study which are probably homogeneous, but also likely to be heterogeneous, finally, every auctioneers can not allow to sell their goods at low price in the well market.

Considering the above, in this paper, we will study how to organize an optimal agenda for the two objects for English auction rule. First, we build a two-object auction model on the basis of the single-object model that described by J.K.Goeree and T.Offerman[14]. Our model considers the common value and private value, the relating factor about the two objects, and without any limitation that a bidder can be winner just once. Second, we study the bidder's equilibrium strategies, the selling price of each individual object, the auctioneer's sum expected revenue of different agendas, then determine optimal agendas. At last, we present two experiments: the first one is used to demonstrate the effectiveness of the analyzed results above; and the second experiment is used to find the changing trend of the optimal agendas.

The remainder of the paper is organized as follows. Section 2 presents the two objects auction settings, describes equilibrium bidding strategies for sequential English auction rule and analyzes the bidding results. Section 3 determines the auctioneer's optimal agendas. Section 4 experiments. Finally, section 5 ends with some conclusions and future work.

## 2 Sequential Auctions

In this paper, on the basis of the model described by J.K.Goeree and T.Offerman [14], considering the common value and private value, we build a two-object auction model. Although S. S. Fatima and M. Wooldridge [13][15][16] extend the single object auction to multi-object auction, they assume winning bidder cannot participate in the following auction, so they think the two objects are independent. But we allow every bidder join in the two auctions, as the result, we must consider the relation between the two objects. First, we review the single auction model built by J.K.Goeree and T.Offerman[14]. Then we present the two objects auction model in this paper and analyze the bidder's equilibrium strategies, the selling price of each individual object, the auctioneer's total expected revenue of different agendas.

### 2.1 Single Object

A single object auction is modeled in [14] as follows. There are  $n \geq 2$  risk neutral bidders. The common value ( $V$ ) of the object to the  $n$  bidders is equal, but initially the bidders do not know this value. However, each bidder receives a signal that gives an estimate of this common value. Bidder  $i = 1, \dots, n$  draws an estimate ( $v_i$ ) from the probability distribution function  $Q(v)$  with support  $[v_L, v_H]$ . Although different bidders may have different estimates, the true value ( $V$ ) is the same for all bidders and is modeled as the average of the bidders' signals:

$$V = \frac{1}{n} \sum_{i=1}^n v_i .$$

Furthermore, each bidder has a cost which is different for different bidders and this cost is its private value. For  $i = 1, \dots, n$ , let  $c_i$  denote bidder  $i$ 's signal for this private value which is drawn from the distribution function  $G(c)$  with support  $[c_L, c_H]$  where  $c_L \geq 0$  and  $v_L \geq c_H$ . Cost and value signals are independently and identically distributed across bidders. Henceforth, we will use term value to refer to common value and cost to private value.

If bidder  $i$  wins the object and pays  $b$ , it gets a utility of  $V - c_i - b$ , where  $V - c_i$  is  $i$ 's surplus, Each bidder bids so as to maximize its utility. Note that bidder  $i$  receives two signals ( $v_i$  and  $c_i$ ) but its bid has to be a single number. Hence, in order to determine their bids, bidders need to combine the two signals into a summary statistic. This is done as follows. For  $i$ , a one-dimensional summary signal, called  $i$ 's surplus, is defined as:

$$S_i = v_i / n - c_i .$$

Suppose  $k$  bidders have dropped out at bid levels  $b_1 \leq \dots \leq b_k$ . A bidder's (say  $i$ 's) strategy is described by functions  $B_k(S_i; b_1 \dots b_k)$ , which specify how high it must bid given that  $k$  bidders have dropped out at levels  $b_1 \dots b_k$  and given that its surplus is  $S_i$ . The  $n$ -tuple of strategies  $(B(\cdot), \dots, B(\cdot))$  with  $B(\cdot)$  are defined as:

$$B_0(x_i) = E(v_i - c_i \mid S_i = x_i)$$

$$B_k(x_i; b_1 \dots b_k) = \frac{n - k}{n} E(v_i \mid S_i = x_i) + \frac{1}{n} \sum_{j=0}^{k-1} E(v_i \mid B_j(S_i; b_1, \dots, b_j) = b_{j+1}) - E(c_i \mid S_i = x_i)$$

Let  $f^n$  denote the first order statistic of the surplus for the  $n$  bidders and let  $s^n$  denote the second order statistic.  $f^n$  and  $s^n$  are obtained from the distribution functions  $Q$  and  $G$ . For the above equilibrium, the bidder with the highest surplus wins. The expected selling price (denoted  $E(P)$ ) is:  $E(P) = E(s^n)$ . The winner's expected profit (denoted  $E(\pi)$ ) is:  $E(\pi) = E(f^n) - E(s^n)$ . The total expected surplus (denoted  $E(W)$ ) is the surplus that gets split between the auctioneer and the winning bidder. It is:  $E(W) = E(V) - E(c)$ . Finally, the expected revenue (denoted

$E(R)$  is the difference between the surplus( $E(W)$ ) and the winner's expected profit( $E(\pi)$ ). This revenue is:  $E(R) = E(W) - E(\pi)$ .

## 2.2 Two Objects

On the basis of the above single-object auction model, we present a two-object auction model, describe the bidder's strategies for sequential English auctions rules, and analyze the selling prices, the winner's profit.

### 2.2.1 Two Objects Model

We define the model as  $M = \langle an, bn, v_1, v_2, c_1, c_2, \alpha, B_1, B_2 \rangle$ . Where,  $an$  is the total number of auctioneers, and  $bn$  is the number of bidders.  $v_1$  denotes the common values estimated by all of the bidders for object1, that is  $v_1 = \langle v_{11}, v_{12}, \dots, v_{1n} \rangle$ , and  $v_2 = \langle v_{21}, v_{22}, \dots, v_{2n} \rangle$  is the values for object2.  $c_1 = \langle c_{11}, c_{12}, \dots, c_{1n} \rangle$  and  $c_2 = \langle c_{21}, c_{22}, \dots, c_{2n} \rangle$  denotes the private value of each bidder of object1 and object2 respectively.  $\alpha = \langle \alpha_1, \alpha_2, \dots, \alpha_n \rangle$  denotes all bidder's estimated relating factors between object1 and object2.  $B_1 = \langle B_1, B_2, \dots, B_n \rangle$  and  $B_2 = \langle B_1, B_2, \dots, B_n \rangle$  are the equilibrium bidding strategies of all of the bidders to the first and second auctions for sequential English auction rule.

This model considers three points:

1. The objects for sale in two auctions display both private and common value characteristics to participants;
2. These two objects are auctioned one after another, and the winner for the first object can participate in the second auction;
3. Although the two objects have same true common values to all bidders, each bidder receives a private value part endogenously. If a bidder can win two objects, it must consider the relation of two objects either complements or substitutes.

**Assumption 1.** There are  $n \geq 2$  risk neutral bidders. Bidder  $i = 1, \dots, n$  receives object1's private cost signal,  $c_{1i}$ , object2's private cost signal,  $c_{2i}$ , and an independent object1's common value signal,  $v_{1i}$ , an independent object2's common value signal,  $v_{2i}$ . Both objects' cost and value signals are assumed to be independently and identically distributed across bidders.

For the sequential English auction rule, assume there are  $n \geq 2$  risk neutral bidders. They receive the private and common value signals for an auction just before that auction begins. Thus, the signals for the second object are received only after the first auction has been conducted. Consequently, although the bidders know the distribution functions from which the signals are drawn, they do not know the actual signals for the second object until the first auction is over. Both in the two auctions, the common value ( $V_1$ ) of the object1 and ( $V_2$ ) of the object2 to the  $n$  bidders is equal respectively. Before the auction is over, the value are unknown for all bidders but can be estimated. Bidder  $i = 1, \dots, n$  draws an estimation( $v_{1i}$ ) of the object1's true

value( $V_1$ ) from the probability distribution function  $Q_1(v)$  with support  $[v_{1L}, v_{1H}]$ . and draws ( $v_{2i}$ ) of the object2's true value( $V_2$ ) from  $Q_2(v)$  with support  $[v_{2L}, v_{2H}]$ . Although different bidders may have different estimations, the true common value ( $V_1$ ) and ( $V_2$ ) are the same for all the bidders. Similar to single object model, the two objects' true common value can be modeled as the average of the bidders' signals:

$$V_1 = \frac{1}{n} \sum_{i=1}^n v_{1i}, \quad V_2 = \frac{1}{n} \sum_{i=1}^n v_{2i},$$

For  $i = 1, \dots, n$  let  $c_{1i}$  denote the  $i$ th bidder's signal for object1's private value which is drawn from the distribution function  $G_1(c)$  with support  $[c_{1L}, c_{1H}]$  where  $c_{1L} \geq 0$ ,  $v_{1L} \geq c_{1H}$ , and  $c_{2i}$  is for object2's private value drawn from  $G_2(c)$  with support  $[c_{2L}, c_{2H}]$  where  $c_{2L} \geq 0$ ,  $v_{2L} \geq c_{2H}$ .

When bidders receive multiple signals they must somehow combine the different pieces of information into a summary statistic since their bid is just a single number. Similar to single object, the  $i$ th surpluses to object1 and object2 are defined as:

$$S_{1i} = \frac{v_{1i}}{n} - c_{1i}, \quad S_{2i} = \frac{v_{2i}}{n} - c_{2i}.$$

To avoid problems of non-monotonicity and non-existence that there is no natural way to order signals in two-dimensions, we shall restrict ourselves to the following class of unimodal densities similar to single object.

**Assumption 2.** The densities  $f_{1v}(v)$ ,  $f_{2v}(v)$ ,  $f_{1c}(c)$  and  $f_{2c}(c)$  are logconcave[17][18].

**Lemma 1.** Under Assumption 2, the conditional expectations  $E(v_1 / S_1 = x)$ ,  $E(v_1 / S_1 \leq x)$ ,  $E(v_2 / S_2 = x)$  and  $E(v_2 / S_2 \leq x)$  are non-decreasing in  $x$ . Furthermore,  $E(c_1 / S_1 = x)$ ,  $E(c_1 / S_1 \leq x)$ ,  $E(c_2 / S_2 = x)$  and  $E(c_2 / S_2 \leq x)$  are non-increasing in  $x$ .

If bidder  $i$  can only win one object, with the analysis above we can deal the case. But when bidder  $i$  has the ability to win both two objects, we have to consider the total common value and the total private value of the two objects.

**Assumption 3.** There are  $n \geq 2$  risk neutral bidders. Bidder  $i = 1, \dots, n$  receives two objects' relating factor signal,  $\alpha_i$ . And it is assumed to be independently and identically distributed across bidders.

Assume there are  $n \geq 2$  risk neutral bidders. They receive two objects' relating factor signal before first auction begins. For  $i = 1, \dots, n$  let  $\alpha_i$  denote the  $i$ th bidder's signal for two objects' relating factor which is drawn from the distribution function  $R(\alpha)$  with support  $[\alpha_L, \alpha_H]$ . Then, for the individual, the total private value will change to:

$$\bar{c}_i = \sum_{j=1}^2 c_{ji} - \alpha_i,$$



Although the total private value is different of the sum of two private values because of the inherent relation of the two objects, the total common value to bidder  $i$  is still the estimate of the common value of object1 adding that of object2:

$$\bar{v}_i = \sum_{j=1}^2 v_{ji},$$

It is because their common value can not be changed regardless of who win them. Thus the total common value of object1 and object2 together can be obtained from the sum estimate of the bidders' two common value:

$$\bar{V} = \frac{1}{n} \sum_{i=1}^n \bar{v}_i = \frac{1}{n} \sum_{i=1}^n \sum_{j=1}^2 v_{ji},$$

Then the surplus of bidder  $i$  winning both the objects is as follow:

$$\bar{S}_i = \frac{\bar{v}_i}{n} - c_i = \frac{\sum_{j=1}^2 v_{ji}}{n} - \sum_{j=1}^2 c_{ji} + \alpha_i$$

In this section, we build a two-object sequential auction model for English auction rule according to our requirements.

**2.2.2 Equilibrium Bidding Strategies**

The two objects are auctioned in two separate English auctions that are conducted sequentially. The English auction rules are as follows. The auctioneer continuously raises the price, and bidders publicly reveal when they withdraw from the auction. Bidders who drop out from an auction are not allowed to re-enter that auction. A bidder's strategy for the first auction depends on how much profit it expects to get from the second auctions yet to be conducted. However, since there are no more auctions after the second. Thus, a bidder's strategic behavior during the second auction is the same as that for a single-object English auction. This has been described by S. S. Fatima[13][15][16]. Equilibrium bidding strategies for a single object of the type described in Section 2.1 have been obtained in [14]. We don't say unnecessary details here. Only due to our model allow the winner for the first object to participate in the second auction, the bidder number who take part in second auction is same as first auction.

**2.2.3 Analysis of Bidding Results**

Before analyzing the selling prices, the winner's profit, we firstly define some concepts: let  $f_1^n = \max S_{1i}$  denote the first order statistic of the surplus of object1 for the  $n$  bidders, let  $s_1^n = \max\{ \{S_{1i}\} - \{f_1^n\} \}$  denote the second order statistic of object1, let  $f_2^n = \max S_{2i}$  denote the first order statistic of object2 and let  $s_2^n = \max\{ \{S_{2i}\} - \{f_2^n\} \}$  denote the second order statistic of object2 respectively.

We know that the selling price of an individual object depends on the auction in which it is sold. There are two possible agendas (1,2) and (2,1) about sequential auctions for the two objects. Let  $E_{12}(\pi_1)$  denote the winner's expected profit of object1

for agenda(1,2), and  $E_{21}(\pi_2)$  denote the winner's expected profit of object2 for agenda(2,1). As the single object case, the winner's expected profit for the first auctions are:

$$E_{12}(\pi_1) = [E(f_1^n) - \pi_2] - [E(s_1^n) - \pi_2] = E(f_1^n) - E(s_1^n),$$

$$E_{21}(\pi_2) = [E(f_2^n) - \pi_1] - [E(s_2^n) - \pi_1] = E(f_2^n) - E(s_2^n),$$

Although the bidders know the distribution (from which the cost and value signals are drawn) before the first auction begins, they draw the signals for the second object only after the first auction ends. So, each of the  $n$  agents that participate in the second auction have an equal chance to win it and the ex ante expected profit are as follows:

$$\pi_2 = \frac{E(f_2^n) - E(s_2^n)}{n}, \quad \pi_1 = \frac{E(f_1^n) - E(s_1^n)}{n},$$

Let  $E_{12}(P_1)$  denote the object1's selling price for agenda (1,2), and let  $E_{21}(P_2)$  denote the object2's selling price for agenda (2,1). Because the bidder's strategies in our model are similar to those in the single object case, the equations for the expected selling price for single object case can be easily adapted to the first auction of the two objects case which is as follows.

$$E_{12}(P_1) = E(s_1^n) - \pi_2, \quad E_{21}(P_2) = E(s_2^n) - \pi_1,$$

When we analyze the second auction result, it may be affected because of the effect of the relating factor  $\alpha_i$ , then if the  $E_{12}(\pi_2)$  denote the winner's expected profit of object2 for agenda(1,2),  $E_{21}(\pi_1)$  denote the winner's expected profit of object1 for agenda(2,1),  $E_{12}(P_2)$  denote the selling price of object2 for agenda(1,2),  $E_{21}(P_1)$  denote the selling price of object1 for agenda(2,1), then:

while  $S_{2i} + \alpha_i > f_2^n$ ,

$$E_{12}(\pi_2) = E(S_{2i} + \alpha_i) - E(f_2^n), \quad E_{12}(P_2) = E(f_2^n),$$

while  $f_2^n > S_{2i} + \alpha_i > s_2^n$ ,

$$E_{12}(\pi_2) = E(f_2^n) - E(S_{2i} + \alpha_i) \quad E_{12}(P_2) = E(S_{2i} + \alpha_i),$$

while  $s_2^n > S_{2i} + \alpha_i$ ,

$$E_{12}(\pi_2) = E(f_2^n) - E(s_2^n), \quad E_{12}(P_2) = E(s_2^n),$$

while  $S_{1i} + \alpha_i > f_1^n$ ,

$$E_{21}(\pi_1) = E(S_{1i} + \alpha_i) - E(f_1^n), \quad E_{21}(P_1) = E(f_1^n),$$

while  $f_1^n > S_{1i} + \alpha_i > s_1^n$ ,

$$E_{21}(\pi_1) = E(f_1^n) - E(S_{1i} + \alpha_i), \quad E_{21}(P_1) = E(S_{1i} + \alpha_i),$$

while  $s_1^n > S_{1i} + \alpha_i$ ,

$$E_{21}(\pi_1) = E(f_1^n) - E(s_1^n), \quad E_{21}(P_1) = E(s_1^n),$$

After getting the different results of agenda (1,2) and (2,1),we will try to study the optimal agendas between two cases.

### 3 The Optimal Agendas

Of course, the sum of the selling prices for the two objects depends on the auction agendas. The agenda for which the sum of the selling prices is the highest among all possible agendas is the auctioneer’s optimal agenda [19]. Here, we determine this agenda.

Through above analysis, we get all objects’ selling prices of different two agendas, then we will determine the optimal. Let  $E_{12}(P)$  denote the cumulative expected selling prices for the two objects for agenda(1,2), and  $E_{21}(P)$  denote the cumulative expected selling prices for the two objects for agenda(2,1). If  $E_{12}(P) - E_{21}(P) \geq 0$ , then the optimal agenda is (1,2). Otherwise, it is (2,1). In order to determine the optimal agenda, we use the notion of dispersion of order statistics [20] since it helps in determining whether  $E_{12}(P) - E_{21}(P)$  is greater than zero or not. The dispersion of order statistics for the surplus is defined as follows:

**Definition 1.** The surplus for object 1 is said to have more dispersed order statistics than object2 if the following condition is true.

$$[E(f_1^n) - E(s_1^n)] > [E(f_2^n) - E(s_2^n)]$$

Here we assume that the dispersion of order statistics of surplus of object1 is higher than that of object2. From Section2.2.3 we can get  $E_{12}(P) - E_{21}(P)$ :

while  $\alpha_i > \max\{f_1^n - S_{i^*}, f_2^n - S_{2i^*}\}$ ,

$$\begin{aligned} E_{12}(P) - E_{21}(P) &= [E_{12}(P_1) + E_{12}(P_2)] - [E_{21}(P_2) + E_{21}(P_1)] \\ &= [E(s_1^n) - \pi_2 + E(f_2^n)] - [E(s_2^n) - \pi_1 + E(f_1^n)] \\ &= [E(s_1^n) - \frac{E(f_2^n) - E(s_2^n)}{n} + E(f_2^n)] - [E(s_2^n) - \frac{E(f_1^n) - E(s_1^n)}{n} + E(f_1^n)] \\ &= \frac{n-1}{n} \{ [E(f_2^n) - E(s_2^n)] - [E(f_1^n) - E(s_1^n)] \} \\ &\quad \because [E(f_2^n) - E(s_2^n)] - [E(f_1^n) - E(s_1^n)] < 0 \\ &\quad \therefore [E_{12}(P_1) + E_{12}(P_2)] - [E_{21}(P_2) + E_{21}(P_1)] < 0 \end{aligned}$$

$\therefore E_{12}(P) - E_{21}(P) < 0$

while  $\alpha_i < \min\{s_1^n - S_{i^*}, s_2^n - S_{2i^*}\}$ ,

$$\begin{aligned} E_{12}(P) - E_{21}(P) &= [E_{12}(P_1) + E_{12}(P_2)] - [E_{21}(P_2) + E_{21}(P_1)] \\ &= [E(s_1^n) - \pi_2 + E(s_2^n)] - [E(s_2^n) - \pi_1 + E(s_1^n)] \\ &= \pi_1 - \pi_2 \\ &= \frac{E(f_1^n) - E(s_1^n)}{n} - \frac{E(f_2^n) - E(s_2^n)}{n} \end{aligned}$$

$$\begin{aligned}
 &= \frac{1}{n} \{ [E(f_1^n) - E(s_1^n)] - [E(f_2^n) - E(s_2^n)] \} \\
 &\because [E(f_1^n) - E(s_1^n)] - [E(f_2^n) - E(s_2^n)] > 0 \\
 &\therefore [E_{12}(P_1) + E_{12}(P_2)] - [E_{21}(P_2) + E_{21}(P_1)] > 0
 \end{aligned}$$

$$\therefore E_{12}(P) - E_{21}(P) > 0$$

$\therefore$  while  $\alpha_i > \max\{f_1^n - S_{1i}, f_2^n - S_{2i}\}$ , it is optimal for the auctioneer to auction objects2 first, and while  $\alpha_i < \min\{s_1^n - S_{1i}, s_2^n - S_{2i}\}$ , to auction objects1 first will get greater cumulative expected selling prices.

while  $\max\{f_1^n - S_{1i}, f_2^n - S_{2i}\} > \alpha_i > \min\{s_1^n - S_{1i}, s_2^n - S_{2i}\}$ ,

we suppose  $E_{12}(P) - E_{21}(P) > 0$

then  $E_{21}(P_1) - E_{12}(P_2) < E_{12}(P_1) - E_{21}(P_2)$

and  $\because E_{12}(P_1) - E_{21}(P_2) = [E(s_1^n) - \pi_2] - [E(s_2^n) - \pi_1]$

$\therefore E_{21}(P_1) - E_{12}(P_2) < [E(s_1^n) - \pi_2] - [E(s_2^n) - \pi_1]$

Because  $E_{21}(P_1)$  and  $E_{12}(P_2)$  are functions of  $S_{1i}$ ,  $\alpha_i$  and  $S_{2i}$ ,  $\alpha_i$  respectively, so we can get function  $F_1$ :

$$F_1(S_{1i}, S_{2i}, \alpha_i) = E_{21}(P_1) - E_{12}(P_2)$$

No matter who win the two objects, their common value will not change immediately, so we can deduce function  $F_2$ :

$$F_2(c_{1i}, c_{2i}, \alpha_i) = F_1(S_{1i}, S_{2i}, \alpha_i)$$

So we get result that while  $F_2 < [E(s_1^n) - \pi_2] - [E(s_2^n) - \pi_1]$ , it is optimal to conduct object1 first; while  $F_2 > [E(s_1^n) - \pi_2] - [E(s_2^n) - \pi_1]$ , it is optimal to conduct object2 first; and while  $F_2 = [E(s_1^n) - \pi_2] - [E(s_2^n) - \pi_1]$ , it is equal to auctioneer in the two agendas.

Our optimal agendas are different of that S.S.Fatima *et al* [13]deduced. Intuitively, this is because their model just think of the second auction's effect beforehand when conducting the first, and we consider the relation of two objects endogenously, when the first object's winner participates in the second, he can calculate his total profit accurately, so the prediction counts for little.

## 4 Experiments and Analysis

In order to find the optimal agendas we make two experiments. In experiment one, it is validated that it may be the optimal strategy either by increasing order or decreasing order of the dispersion of the order statistics of the surpluses. In experiment two, the changing trend of the optimal agenda, which is influenced by the relation of the objects and between the dispersion of the order statistics of the surpluses of the bidder, is found.

In experiment one, we use the auction model building on the basis of the mobile agent system Aglet of IBM to make five-groups experiments(<stock of electric plant, stock of aluminum plant>,<use of channel one, use of channel two>,<house one,

house two>,<house, basement of the same section>,<house, shop nearby>), two objects for each group. The relation is determined by relating factor  $\alpha$ , which should be designed to the ratio of exchangeable value of the two objects and the sum of the two original private values. For the sake of convenience, in this experiment, it is directly used to stand for the exchangeable value of the two objects. There are ten persons for each group, and the bidders send both the private value and the common value signals of the two objects to bidders' agent. Then the auctioneers send the common value, private value and  $\alpha$  signals that they think experientially to auctioneer agent. According to the ten bidders, the auctioneer agent finds the optimal agenda firstly. And then organizes two auctions for each group objects. The experimental result shows that the optimal agenda can increase the earning of the auctioneer by 1%-1.5% versus the other agenda. In addition, the objects in first, 4th and 5th groups are complements each other. The total price of them is higher when first bid the object with lower dispersion of the order statistics of the surpluses.

In experiment two, the  $2 \times 10^4$  groups of bidder's surpluses and the relating factor  $\alpha$  with two objects are created using JAVA. All the datum distributes into the scale of  $y(0 \sim 10)$ , which is the ratio of the dispersion of the order statistics of the surpluses of object1 and that of object2, and the scale of  $x(0 \sim 10)$ , which is the ratio of total private value of the two objects and the sum of the two original private value. Then we use the auction simulating system built in experiment 1 to conduct two kinds of agendas one by one, obtaining the ratio of auctioneer's revenue of agenda(1,2) and that of agenda(2,1). As described below the changing trend of  $z$  along with  $x$  and  $y$  can be presented directly, when  $z < 1$ , the auctioneer's optimal agenda is agenda(2,1) that is, it is optimal for the auctioneer to auction objects in the decreasing order of the dispersion of the order statistics of the surpluses.

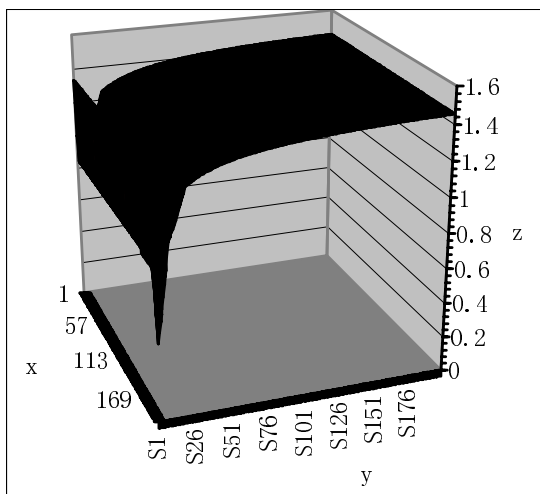


Fig. 1. The ratio of two agendas' profits

## 5 Conclusions and Future Work

This paper has analyzed sequential English auctions for heterogeneous objects with private and common values in an incomplete information setting. We first present a two objects sequential English auctions model, then analyze equilibrium strategies for each bidder in sequence. On the basis of these equilibrium, we determine the winner's profit and expected selling prices of each auction in all agendas, deduce optimal agendas. We show that, optimal agenda may be either by increasing order or decreasing order of the dispersion of the order statistics of the surpluses. Finally, we make two experiments, one validates the result above, and the other display the changing trend of the optimal agenda.

For the future, we will analyze the contour line of optimal agenda accurately (finding the function  $F_2$ ), and study the case for more than two objects for other auction rules.

## References

1. Cassady R, Auctions and Auctioneering. Berkeley: University of California Press. 1967[M]
2. McAfeeRP, JMcMillan. Auction sand Bidding. Journal of Economic Literature 25(2)699-738. 1987[J].
3. William Vickrey, Counterspeculation, Auctions, and Competitive Sealed Tenders. The Journal of Finance, 1961
4. Riley J.Samuelson W. Optimal auctions[J]. American Economic Review 1981, 71:381-92.
5. S.J.Rassenti,V.L.Smith,R.L.Bulfin.A combinatorial auction mechanism for airport time slot allocation[J].Bell Journal of Economics,1982,13:402~417
6. Erthan Kutanoglu,S.David Wu.On combinatorial auction and lagrangean relaxation for distributed resource scheduling[J].IEEE Transactions,1999,813~826
7. T.Sandholm and S.Suri. BOB:Improved winner determination in combinatorial auctions and generalizations.Artificial Intelligence,145:33-58,2003
8. Maskin,Eric and John Riley.Optimal Multi-unit auctions.The Economics of Missing Markets,Information,and Games[M].Frank hahn(ed.),Oxford university Press,1989
9. Maskin,Eric and John Riley. Asymmetric Auctions[J].Review of Economic Studies,2000,67:413~438
10. W.Elmaghraby.The importance of ordering in sequential auctions.Management Science,49(5):673-682,2003.
11. Jean-Pierre Benoit and Vijay Krishna. Multiple-Object Auctions with Budget Constrained Bidders.Review of Economic Studies,68:155-179,2001
12. Mireia Jofre-Bonet and Martin Pesendorfer, Optimal Sequential Auctions, technical report, LSE and submitted to a peer reviewed journal, May 2005
13. S. S. Fatima, M. Wooldridge, and N. R. Jennings. Optimal Agendas for Sequential Auctions for Common and Private Value Objects In Proceedings of the 2005 Workshop on Game Theory and Decision Theory in Agent Systems (GTDT-05), Edinburgh, UK, July 2005
14. J.K.Goeree and T.Offerman.Competitive bidding in auctions with private and common values.The Economic Journal,113(489):598-613,2003.

15. S. S. Fatima, M. Wooldridge, and N. R. Jennings. Sequential Auctions for Objects with Common and Private Values. In Proceedings of the Fourth International Joint Conference on Autonomous Agents and Multi-Agent Systems (AAMAS-05), pp 635-642, , Utrecht, the Netherlands, July 2005
16. S. S. Fatima, M. Wooldridge, and N. R. Jennings. An Analysis of Sequential Auctions for Common and Private Value Objects. In Proceedings of the Seventh International Workshop on Agent-Mediated Electronic Commerce (AMEC-05), pp 25-38, Utrecht, The Netherlands, July 2005
17. Mark Bagnoli and Ted Bergstrom. Log-concave probability and its applications. *Economic Theory* RePEc:spr:joecth:v:26:y:2005:i:2:p:445-469
18. Mark Y An. Logconcavity versus Logconvexity: A Complete Characterization. *JOURNAL OF ECONOMIC THEORY*, Vol. 80, 1998, pages 350-369
19. R.B. Myerson. The basic theory of optimal auctions. In Engelbrecht-Wiggans R, M. Shubik, and R.M. Stark, editors, *Auctions, Bidding, and Contracting: Uses and Theory*, pages 149-164. New York University Press, 1983.
20. B. Khaledi and S. Kochar. On dispersive ordering between order statistics in one-sample and two-sample problems. *Statistics and Probability Letters*, 46:257-261, 2000.

# Intelligent Game Agent Based Physics Engine for Intelligent Non Player Characters\*

Jonghwa Choi, Dongkyoo shin, and Dongil Shin\*\*

Department of Computer Science and Engineering, Sejong University,  
98 Kunja-Dong Kwangjin-Gu, Seoul, Korea  
jhchoi@gce.sejong.ac.kr, {shindk, dshin}@sejong.ac.kr

**Abstract.** This paper presents the intelligent game agent that gives effective intelligence to NPCs (Non Player Characters) for which intelligence did not exist. Generally, non-player characters (NPCs), or agents, such as monsters, enemy guards, or friendly wingmen, can be controlled by a finite-state machine. To overcome the shortcoming of NPC's restricted action, we applied a LSVM (Linear Support Vector Machine) as pattern recognition for the intelligent game agent, and processed all data in XML format to handle the data efficiently. The intelligent agent is executed in the base of the game physics engine. A lot of NPCs that act in a game learn physics values that are produced in the game, and change NPS's action intelligently. We applied two pattern recognition algorithms to estimate the algorithm's performance through comparison. As indicated by experiments, when the M-BP has a fixed number of input layers (number of physical parameters) and output layers (impact value), it shows the best performance when the number of hidden layers is 3 and the learning count number is 30,000. The pattern recognizer that applied LSVM shows the best performance when the learning count number is 25000, and the LSVM shows better performance than the M-BP in the intelligent game agent.

**Keywords:** Non-Player Game Character, Physics Game Engine, AI Game Engine.

## 1 Introduction

The value of the world's game market was, in round figures, 56.1 billion dollars in 2004, and showed in 2005 an increase of 0.8% as compared with the preceding year. It is predicted that the growth rate will average more than 10% in 2006 and 2007 [1]. Continuous growth of the game market is proof that game is an important element in a human's life. 3D games are recognized as an essential element of game manufacture and depend on the development of hardware. The most important issue for a 3D game is a study of the motion and intelligent action of the game characters. And when a game includes these functions, it must not experience performance problems. The motion of a game character, including-its position and displacement, should be calculated based on physical laws; a physics engine takes charge of this role [2].

---

\* This study was supported by a grant of the Seoul R&BD Program.

\*\* Corresponding author.



A physics engine only expresses realistic motion for a character. Intelligent action by a game character is expressed by the game's artificial intelligence engine. However, many researchers have studied poor characters that act according to limited rules in the game world. This paper presents the intelligent game agent that give an effective intelligence to NPCs (Non Player Characters) in which intelligence doesn't exist. Generally, non-player characters (NPCs), or agents, such as monsters, enemy guards, or friendly wingmen, can be controlled by a finite-state machine [3].

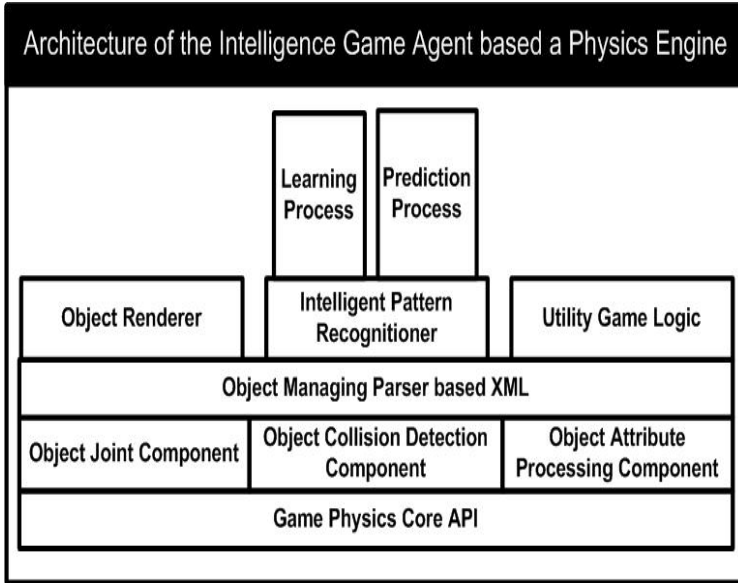
To overcome restricted action of NPC, we applied a LSVM (Linear Support Vector Machine) as the pattern recognition method of the intelligent game agent, and processed all data in XML format to handle the data efficiently. The intelligent agent is executed in the base of the game physics engine. A lot of NPCs that act in game, learn physics values that are produced in game, and influences the NPS's action intelligently. Section 2 present previous studies related to implementing physics engines and artificial intelligence in games. Section 3 explains the architecture of the intelligent game agent that predicts action of NPCs. Section 4 presents algorithm of the intelligent game agent and inner process's data structure. Section 5 shows demo simulation and experimental result. We conclude with section 6.

## 2 Related Research Studies

The physics engine has been developed by many game researchers. Physics engines are offered as commercial physics engines and open source physics engines. Math Engine [4], Havok [5], and Meqon [6] are widely used commercial physics engines. Also, ODE (Open Dynamics Engine) is a well-known open source physics engine [7]. The study that led to a physical model for the solution of physical problems in the real world also led to an algorithm for engine structure [2]. The interaction of multiple parts in the execution of the physiotherapy engine developed from the study of collision detection; one such study led to *voxel's* efficient structure to improve the speed of collision detection [8]. Research that compared performance of collision detection method algorithms was described in the *boxtree* method that is restricted by structure of object [9]. Research on artificial intelligence for physics situation recognition is incomplete. A game's AI SDK recently presented are DirectIA (made by the Mathematiques Appliquees), Spark (made by Louder Than A Bomb), Memetic Artificial Intelligence Toolkit (made by memeticai.org) and RenderWare AI SDK (made by Cirterion) [10-13]. In a detailed study of game AI, Wen Tang simulated intelligent self-learning characters for computer games or other interactive virtual applications. The complex learning behaviors of the virtual characters are modeled as an evolutionary process so that adaptive AI algorithms such as genetic algorithms have been used to simulate the learning process [14]. Yee Chia Hui presented an AI game structure using a path finding and A\* search algorithm [15]. Richard Cant described a hybrid artificial intelligence approach combining soft AI techniques (neural networks) and hard AI methods (alpha-beta game tree search), in an attempt to approximate human play more accurately, in particular with reference to the game of Go [16]. As a contribution to a field that isn't studied until presently, the intelligent game agent that is presented in this paper takes charge of a role that offers game intelligence to NPCs using the learning of physics values that are produced from game characters.

### 3 Architecture of the Intelligent Game Agent

Figure 1 shows the architecture of the intelligent game agent. The intelligent game agent that is presented in this paper can be applied to a FPS (first person shooting) game that is made 3D.



**Fig. 1.** Architecture of the intelligence game agent based a physics engine

Because the intelligent game agent decides the action of all NPCs that use physics values that occur in a game, physics engine is essential for it. The game intelligent agent classifies the core API of the physics engine into components of three kinds (the object joint component, the object collision detection component and the object attribute processing component). The object attribute-processing component manages the attributes of all objects that use the physics attributes of game characters (position vector, linear velocity, mass and etc). The object joint component sets up the joint relations of physics objects, and takes charge of the role that changes values of position and rotation in objects. The object collision detection component controls object’s moving through collision detection among objects. Three components are executed dependently on each other. The core of physics object calculates moving distance and rotation value of each object, and expresses it to game world. All physics values are managed in the object’s attribute-processing component, and the object’s attribute-processing component calculates the accurate attribute’s value using the object joint component and the object collision detection component. We manage the object structure and attribute’s value that existed in the game as a XML tree [17]. The data structure of the XML format has the advantage of efficiency of data management and integration with different physics engines. The object managing parser converts all objects into the XML tree, and offers easy access of objects to three components

(the object renderer, the intelligent pattern recognizer, the utility game logic). The intelligent pattern recognizer learns patterns of impact values (user character and NPC) and physics values of objects, and informs the time that NPC give maximum impact to user character. Impact means the value that is produced when the user character (that is controlled by user) and NPSs (that makes an attack against on the user character) collides. The object renderer acquires physics values of all objects in XML tree and expresses all objects in game world. The utility game logic takes charge of level design in the game.

### 4 Description of Detailed Components

Figure 2 shows the game physics API and relation among all physics components.

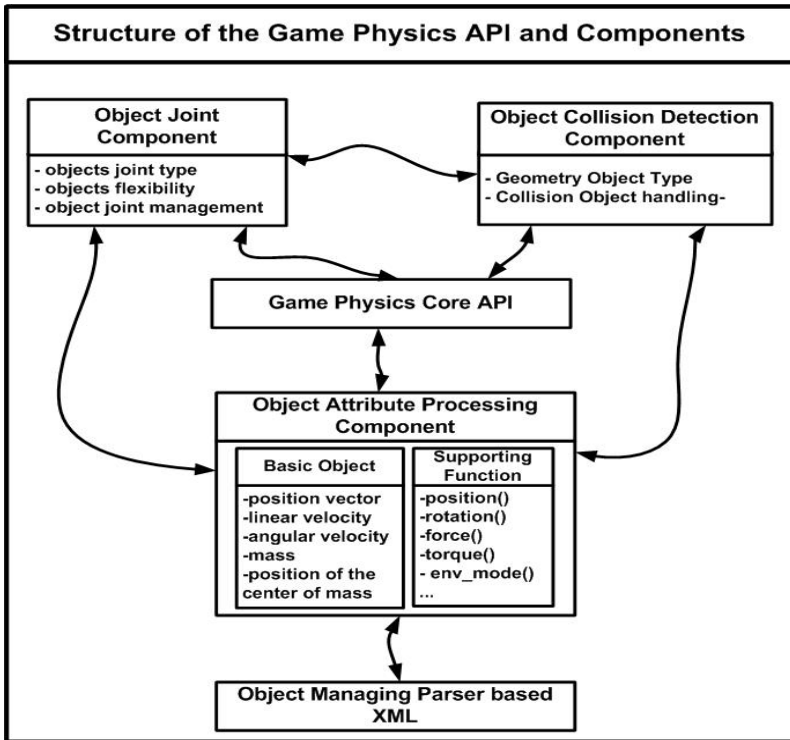
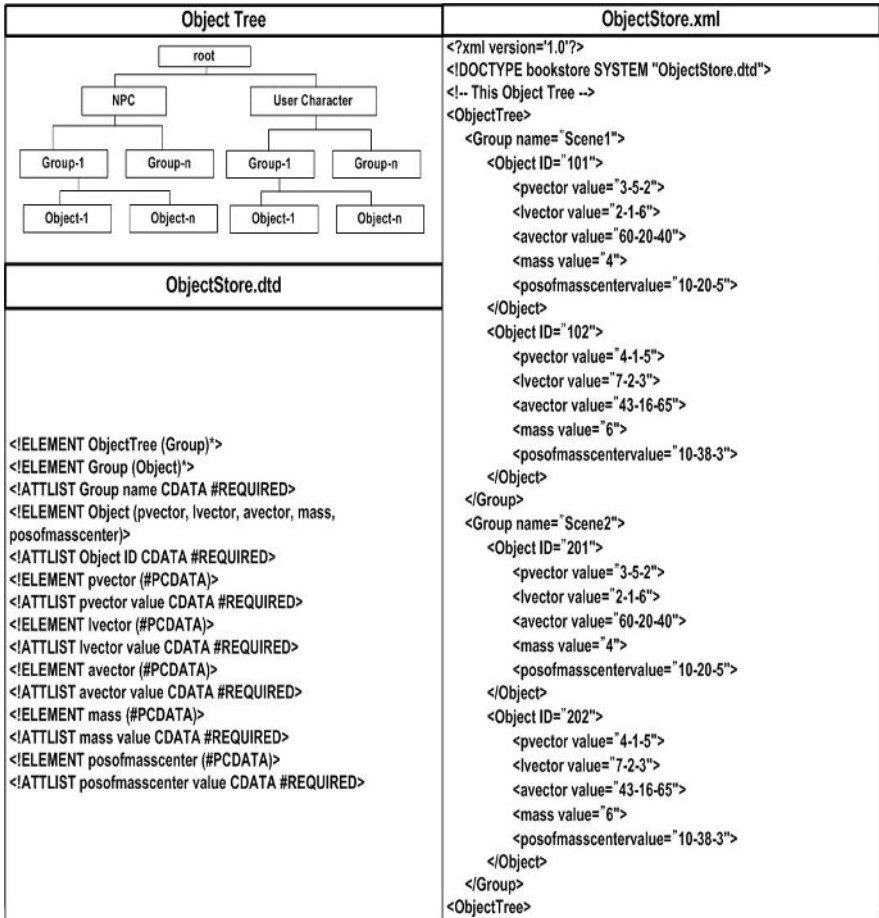


Fig. 2. Structure of the game physics API and components

The intelligent game agent uses six physics values to predict intelligent action of NPCs. Figure 2 shows three components that calculate six physics values of the object. The game physics core API defines the basic data type of the physics object and math function for physics calculation. The object attribute-processing component includes six physics values (position vector, linear velocity, angular velocity, mass and position of the center of mass), and calculates moving distance and rotation

values using these values. The six values that are included in the object’s attribute processing are applied as input values for intelligent pattern recognizer. The object collision detection component and the object joint component calculate the relation of the object’s binding and rotation through interaction. The object managing parser creates an object tree in XML format that includes all objects, and changes attributes of objects using the object attribute-processing component. Figure 3 shows the object tree in XML format, DTD file and XML file.



**Fig. 3.** Structure of Object tree and ObjectStore.dtd and ObjectStore.xml that is used in the object managing parser

The object tree defines the root as the top-level element. The root element includes user characters and NPC elements as child element. The NPC and the user character have a group element as child element, and the group element has an object element that exists in game. The object element has a unique ID, and includes six physics values. If manages all objects as a XML tree, it is effective in search and change of

object and structure’s conversion of object’s relation. Because the object tree in XML format exists in memory, access speed of other process is fast. Also, as we applied the XML tree in object management, porting of other physics engines are easy in the intelligent game agent. To apply other physics engines in the intelligent game agent, the interface is decided according to a prescribed file (objectStore.dtd).

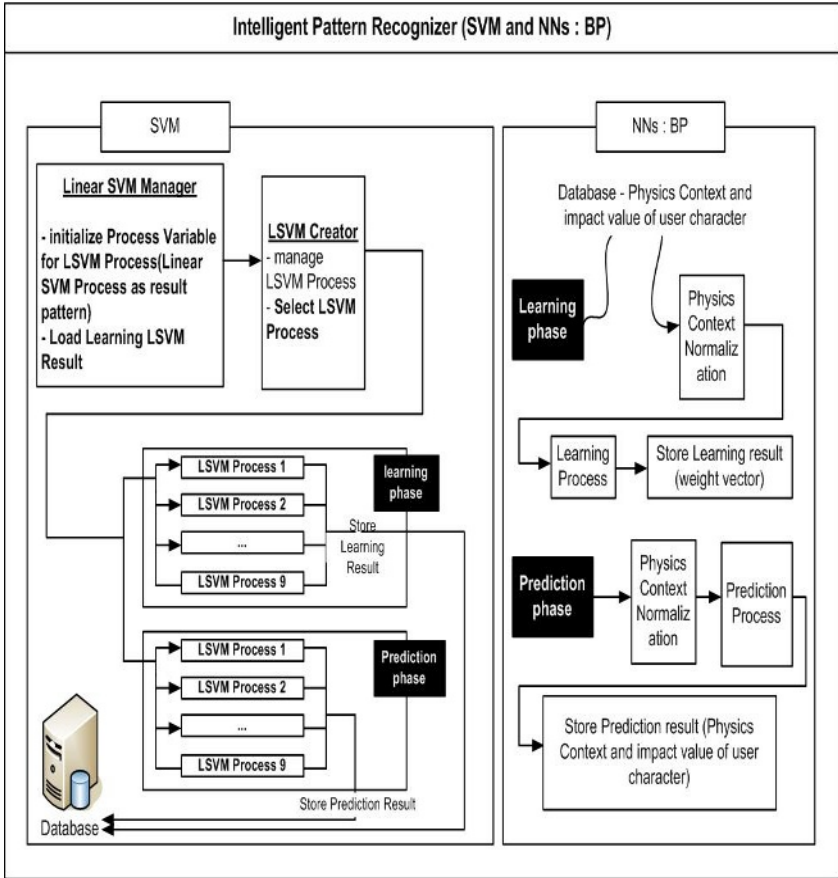


Fig. 4. Structure of the intelligent pattern recognizer

If user character and NPC collide, the intelligent pattern recognizer learns the physics pattern using the impact value (of NPC and user character) and six physics values (of NPC and user character), and periodically predicts the impact value between user character and NPC that is near to the user character, and decides the action of the NPC through comparison of the NPC’s impact value and the user character’s impact value. We applied the LSVM as pattern recognition algorithm to analyze the pattern of six physics values [18]. Figure 4 shows structure of the intelligent pattern recognizer that applied two pattern recognition algorithms. Additionally, we applied the M-BP (momentum back-propagation) algorithm to compare the intelligent pattern recognizer’s performance [19]. Because we sorted the

impact value of a character with 9 sectors, the intelligent pattern recognizer applied 9 LSVMs hierarchically. When the intelligent pattern recognizer starts, the linear SVM manager sets up results of the learning algorithm that are acquired from database as initial value. The LSVM creator creates 9 LSVMs hierarchically by sector of impact value. The Inner value of the algorithm that is changed by the LSVM is stored in the database if the intelligent pattern recognizer stops. The learning process starts if two characters (user character and NPC) collide, and the prediction process periodically executes at a frequency that is defined in the intelligent pattern recognizer. Also, figure 4 shows a diagram of M-BP that is applied to compare the intelligent pattern recognizer's performance. The M-BP acquires a weight vector from the database if the intelligent pattern recognizer starts, and stores a weight vector that is changed by the learning process if the intelligent game agent stops. Figure 5 shows the process diagram of the intelligent game agent.

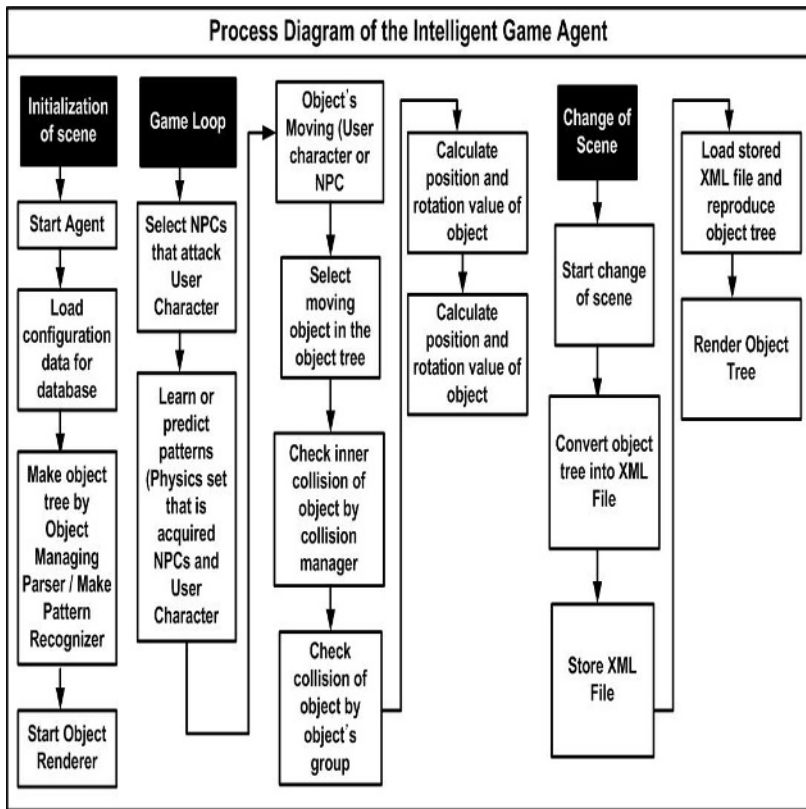


Fig. 5. Process diagram of the intelligent game agent

When the intelligent agent starts, it acquires the total object's structure and the physics value of object from database using the ObjectStore.xml file to create the object tree in XML format. The object managing parser completes the object tree that is used in the game. The object tree that is created, applies the physics values of the

object that is changed by object moving and rotation. The intelligent pattern recognizer creates 9 LSVMs hierarchically by sector of impact value. The object renderer expresses all objects using the object tree in the game world. After the intelligent game agent starts, all processes in the game are run by the game loop. Figure 5 presents a flowchart of the process in which the NPC attacks the user character. Because the intelligent agent applies the total structure based on XML format, a change of scene in the game is easy. The change of scene shows processes in which all game characters are converted, keeping object's structure and attribute.

### 5 Demo Simulations and Experiments

Figure 6 shows a demo simulation of the intelligent game agent. The user character and NPC are managed in the object tree, and NPC attacks the user character if the NPC predicts (by prediction of the intelligent pattern recognizer) that its own situation can give the user character effective damage.

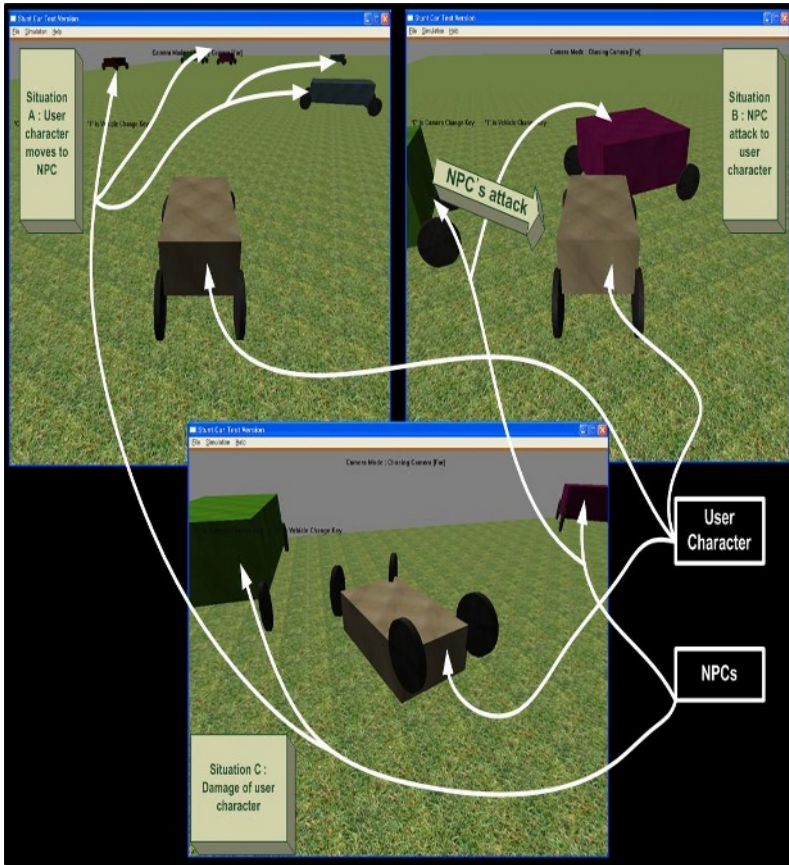


Fig. 6. Demo simulation of the intelligent game agent

We have shown the performance of the intelligent pattern recognizer through experiment. Input values for the pattern recognition algorithm are normalized between 0.1 and 0.9 to heighten algorithm’s performance. Table 1 shows input values that are normalized (physics value).

**Table 1.** Physical Parameters of user character and NPC. <P: position of user character and NPC, L:linear velocity, C:center of mass, M:car’s mass, V:variable velocity>.

Norm Value	User Character					Non Play Character				
	P	L	C	M	V	P	L	C	M	V
0.1	1	0-10	0.0-1.0	1	1-5	1	0-10	0.0-1.0	1	1-5
0.2	2	11-20	1.1-2.0	2	6-10	2	11-20	1.1-2.0	2	6-10
0.3	3	21-30	2.1-3.0	3	11-15	3	21-30	2.1-3.0	3	11-15
0.4	4	31-40	3.1-4.0	4	16-20	4	31-40	3.1-4.0	4	16-20
0.5	5	41-50	4.1-5.0	5	21-25	5	41-50	4.1-5.0	5	21-25
0.6	6	51-60	5.1-6.0	6	26-30	6	51-60	5.1-6.0	6	26-30
0.7	7	61-70	6.1-7.0	7	31-35	7	61-70	6.1-7.0	7	31-35
0.8	8	71-80	7.1-8.0	8	36-40	8	71-80	7.1-8.0	8	36-40
0.9	9	81-90	8.1-9.0	9	41-45	9	81-90	8.1-9.0	9	41-45

**Table 2.** Impact value of user character and NPC. Impact value is divided in 9 sectors.

Impact value of Master Car	0-10	11-20	21-30	31-40	41-50	51-60	61-70	71-80	81-90	91-100
Norm Value	0.0	0.11	0.21	0.31	0.41	0.51	0.61	0.71	0.81	0.91
	-	-0.2	-0.3	-0.4	-0.5	-0.6	-0.7	-0.8	-0.9	-
	0.1									0.99

We applied two pattern recognition algorithms to estimate the algorithm’s performance through comparison. Table 3 shows the performance of the intelligent pattern recognizer that applied M-BP. In an experiment, when the M-BP has a fixed number of input layers (number of physical parameters) and output layers (impact value), it shows the best performance when the number of hidden layers was 3 and the learning count number was 30,000.



**Table 3.** Performance of intelligent pattern recognizer that is applied M-BP

Hidden Layer	Success Rate(%)	Cross Validation error value hidden layer	Cross validation error value by output layer	Test error signal value by output layer
1	63	31.23532	231.34353	234.12622
3	92	13.23243	164.23213	166.09230
5	84	24.12153	128.64009	130.25030
Learning Count	Success Rate(%)	Cross Validation error signal value by hidden layer	Cross validation error signal value by output layer	Test error signal value by output layer
10000	71	28.23553	224.23242	225.55020
20000	81	25.25939	130.96954	131.08834
30000	92	13.23243	164.23213	166.09230
40000	85	23.25534	127.94204	128.03253
50000	85	23.25254	127.89042	128.01223

**Table 4.** Performance of intelligent pattern recognizer that is applied LSVM

Number of learning data	number support vector	norm of longest vector	number of kernel evaluations	precision on test set
1000	52	1.79828	14092	82.32%
5000	78	1.89232	14394	87.24%
10000	98	1.83277	13020	91.87%
15000	92	2.13929	13729	94.11%
20000	101	2.02345	12494	96.54%
25000	129	2.34502	11945	98.67%
30000	110	2.31220	12949	97.12%
35000	122	2.12965	13929	97.14%

Table 4 shows Table 3 shows the performance of the intelligent pattern recognizer that is applied LSVM.

According to these experiments, the pattern recognizer that applied LSVM shows the best performance when the learning count number is 25000, and the LSVM shows better performance than the M-BP in the intelligent game agent.

## 6 Conclusions

This paper presents the intelligent game agent that gives effective intelligence to NPCs (Non Player Characters) for which intelligence did not exist. Generally, non-player characters (NPCs), or agents, such as monsters, enemy guards, or friendly wingmen, can be controlled by a finite-state machine. To overcome the shortcoming of NPC's restricted action, we applied a LSVM (Linear Support Vector Machine) as pattern recognition for the intelligent game agent, and processed all data in XML format to handle the data efficiently. The intelligent agent is executed in the base of the game physics engine. A lot of NPCs that act in a game learn physics values that are produced in the game, and change NPS's action intelligently. We applied two pattern recognition algorithms to estimate the algorithm's performance through comparison. As indicated by experiments, when the M-BP has a fixed number of input layers (number of physical parameters) and output layers (impact value), it shows the best performance when the number of hidden layers is 3 and the learning count number is 30,000. The pattern recognizer that applied LSVM shows the best performance when the learning count number is 25000, and the LSVM shows better performance than the M-BP in the intelligent game agent.

## References

1. <http://www.dfcint.com>: DFC Intelligence, (2002-2004)
2. Kook, H.J, Novak, G. S., Jr.: Representation of models for solving real world physics problems. Proceedings of the Sixth Conference on Artificial Intelligence for Applications, (1990) 274-280
3. Cass. S.: Mind games. Spectrum. IEEE. volume 39. Issue 12. (2002) 40-44
4. Math Engine.: <http://www.mathengine.com>
5. Havok.: <http://havok.com>
6. Meqon.: <http://www.meqon.com>
7. Open Dynamics Engine.: <http://ode.org>
8. Lawlor, O.S., Kalee, L.V.: A Voxel-based Parallel Collision Detection Algorithm. Proceedings of the 6th international conference on Supercomputing, June (2002) 285-293
9. Zachmann, G.: Minimal Hierarchical Collision Detection. Proceedings of the ACM symposium on Virtual reality software and technology, November (2002)
10. DirectIA.: <http://www.masa-sci.com/>
11. Spark.: <http://www.louderthanabomb.com/>
12. Memetic.: <http://www.memeticai.org/>
13. Renderware AI SDK.: <http://www.renderware.com/>
14. Wen Tang, Tao Ruan Wan.: Intelligent self-learning characters for computer games. Eurographics UK Conference 2002 Proceedings. (2002) 51-58
15. Hui. Y.C., Prakash. E.C., Chaudhari. N.S.: Game AI: Artificial intelligence for 3D path finding. TENCON 2004. 2004 IEEE Region 10 Conference. Volume B. (2004) 306 - 309
16. Cant, R. Churchill, J. Al-Dabass, D.: A hybrid artificial intelligence approach with application to games. Neural Networks. Proceedings of the 2002 International Joint Conference on. Volume 2. (2002) 1575 - 1580
17. Phansalkar, V,V. Sastry, P,S.: Analysis of the back-propagation algorithm with momentum. Nerual Networks. IEEE transaction on. (1994) 505-506
18. Burges. C,J,C.: A tutorial on support vector machines for pattern recognition. Data Mining Knowl, Disc. Vol. 2, no. 2. (1998) 1-47

# Palmprint Recognition Based on Improved 2DPCA

Junwei Tao, Wei Jiang, Zan Gao, Shuang Chen, and Chao Wang

School of Information Science & Engineering, Shandong University, Jinan  
taojunwei@mail.sdu.edu.cn

**Abstract.** Palmprint recognition received many researchers' attention because of its low resolution and cheap devices. As other biometrics, algebraic feature is the prevailing method for palmprint recognition. PCA (principal component analysis) is one prevailing algebraic transformation. It has been a successful feature detection method for pattern recognition. It deals with image vector whose dimension is usually high. 2DPCA is a novel PCA method for image matrix, and it can calculate the covariance matrix more precise. In this paper we apply the new 2DPCA method to palmprint recognition, and we make an improvement in the selection of principal components. In our method we select the principal component that is better for classification. At last we do the improved 2DPCA on the row and column direction to reduce dimension in both direction. Then we apply the method to PolyU Palmprint Database. The experiment result shows that our method got more recognition rate with lower dimensions.

## 1 Introduction

Palmprint recognition is one kind of biometric technology. Compared with other biometrics, it is well known for several advantages such as stable line features, low-resolution imaging can be employed; low cost capture devices can be used; it is very difficult to fake a palmprint; the line features of the palmprints are stable, etc. As other biometrics there are many approaches for palmprint recognition in various literatures; most of which are based on structural feature, statistical feature and algebraic feature. However, structural feature such as principal lines [1], wrinkles, delta points, feature points and interesting points are difficult to be extracted, represented and compared while the discriminability of statistical feature such as texture energy [2] is not strong enough for palmprint recognition. Algebraic features can overcome these disadvantages and is the best and prevailing method for the moment.

Principal component analysis, also known as Karhunen Loeve expansion, is a classical feature extraction and data representation technique widely used in the area of pattern recognition and computer vision. In the PCA-based recognition technique, the 2D image matrices must be previously transformed into 1D image vectors. The resulting image vectors usually lead to a high dimensional image vector space, where it is difficult to evaluate the covariance matrix accurately due to its large size and the relatively small number of training samples. Fortunately, the eigenvectors can be calculated efficiently using the SVD techniques [3],[4] and the process of generating

the covariance matrix is actually avoided. However, this does not imply that the eigenvectors can be evaluated accurately in this way. Because the covariance matrix statistically determines the eigenvectors, no matter what method is adopted for obtaining them. Two-dimensional principal component analysis (2DPCA), is developed for image feature extraction. As opposed to conventional PCA, 2DPCA is based on 2D matrices rather than 1D vectors. That is, the image matrix does not need to be previously transformed into a vector. Instead, an image covariance matrix can be constructed directly using the original image matrices. In contrast to the covariance matrix of PCA, the size of the image covariance matrix used in 2DPCA is much smaller. As a result, 2DPCA has two important advantages [5] over PCA. First, it is easier to evaluate the covariance matrix accurately. Second, less time is required to determine the corresponding eigenvectors. The theory of 2DPCA is introduced in section 2.1. And in section 2.2 we will discuss the relation between 2DPCA and line-based PCA.

As we all know, 1DPCA and 2DPCA algorithm select the  $m$ -largest principal components. This selection method is the best for dimension compression, but we have proved that it is not the best for pattern recognition. Then how to select the principal components that are better for pattern recognition? This paper introduced a method to select principal components that are the best for pattern recognition. The selection method is introduced in section 2.3.

At last, we apply our method to PolyU palmprint database, and the experiment result is shown in section 3.

## 2 2DPCA

### 2.1 Idea and Algorithm of 2DPCA

Let  $X$  denotes an  $n$ -dimensional unitary column vector. Our purpose is to project image  $A$ , an  $m \times n$  random matrix, onto  $X$  by the following linear transformation:

$$Y = AX \tag{1}$$

Thus, we obtain an  $m$ -dimensional projected vector  $Y$ , which is called the projected feature vector or principal component of image  $A$ .

How do we determine a good projection vector  $X$ ? In fact, the total scatter of the projected samples can be used to measure the discriminatory power of the projection vector  $X$ . The total scatter of the projected samples can be characterized by the trace of the covariance matrix of the projected feature vectors. From this point of view, we adopt the following criterion:

$$J(X) = \text{tr}(S_X) \tag{2}$$

where  $S_X$  denotes the covariance matrix of the projected feature vectors of the training samples and  $\text{tr}(S_X)$  denotes the trace of  $S_X$ . The physical significance of maximizing the criterion in (2) is to find a projection vector  $X$ , onto which all samples are projected, so that the total scatter of the resulting projected samples is maximized. The covariance matrix  $S_X$  can be denoted by

$$\begin{aligned}
 S_x &= E[(Y - EY)(Y - EY)^T] \\
 &= E\{[AX - E(AX)][AX - E(AX)]^T\} \\
 &= E\{[(A - EA)X][(A - EA)X]^T\}
 \end{aligned} \tag{3}$$

so,

$$tr(S_x) = X^T E[(A - EA)^T (A - EA)]X \tag{4}$$

Let us define the following matrix

$$S_t = E[(A - EA)^T (A - EA)] \tag{5}$$

The matrix  $S_t$  is called the image covariance (scatter) matrix. It is easy to verify that  $S_t$  is an  $n \times n$  nonnegative definite matrix from its definition. We can evaluate  $S_t$  directly using the training image definition. Suppose that there are  $M$  training image samples in total, the  $j$ th training image is denoted by an  $m \times n$  matrix  $A_j$  ( $j=1,2, \dots, M$ ), and the average image of all training samples is denoted by  $\bar{A}$ . Then,  $S_t$  can be evaluated by

$$S_t = (1/M) \sum_{j=1}^M (A_j - \bar{A})^T (A_j - \bar{A}) \tag{6}$$

Alternatively, the criterion in (2) can be expressed by

$$J(X) = X^T S_t X \tag{7}$$

Where  $X$  is a unitary column vector. This criterion is called the generalized total scatter criterion. The unitary vector  $X$  that maximizes the criterion is called the optimal projection axis. Intuitively, this means that the total scatter of the projected samples is maximized after the projection of an image matrix onto  $X$ .

The optimal projection axis  $X_{opt}$  is the unitary vector that maximizes  $J(X)$ , i.e. the eigenvector of  $S_t$  corresponding to the largest eigenvalue. In general, it is not enough to have only one optimal projection axis. We usually need to select a set of projection axes,  $X_1, \dots, X_d$ , subject to the orthonormal constraints and maximizing the criterion  $J(X)$ , that is,

$$\{X_1, \dots, X_d\} = \arg \max(J(X)) \tag{8}$$

In fact, the optimal projection axes,  $X_1, \dots, X_d$ , are the orthonormal eigenvector of  $S_t$  corresponding to the first  $d$ -largest eigenvalues.

## 2.2 Equivalence of 2DPCA to Line-Based PCA

As we all know that the procedure of image line based PCA is as follows. First, partition each image whose resolution is  $m \times n$  into  $m$  image lines. Each image line is considered as a Euclidean vector of dimensionality  $n$ . Then, the  $M$  images in the sample set are decomposed to  $M \cdot m$  vectors. Then PCA is applied on these vectors, i.e., computing the leading eigenvectors of the covariance matrix of the  $M \cdot m$  data vectors, we obtain the line-based PCA features. Then what is the relation between

2DPCA and line-based PCA? Intuitive observation tells us that 2DPCA is very like line-based PCA. Now we will prove that 2DPCA is equivalent to image line-based PCA. For a detailed proof, please see [6]. The main idea is that the image covariance matrix  $S_t$  of 2DPCA is exactly the ordinary sample covariance matrix of all the  $M$ - $m$  image line considered as vectors. More precisely, suppose that  $A_1, A_2, \dots, A_M$  are image matrices of size  $m \times n$ . Let  $a_{i1}, a_{i2}, \dots, a_{im}$  be the  $m$  lines of  $A_i$ , and we will assume that, without loss of generality, data matrices have been shifted so that they have zero mean, i.e.,  $\bar{A} = (1/M) \sum_{i=1}^M A_i = 0$ , accordingly, the image covariance matrix

reduces to a simpler form  $S_t = \frac{1}{M} \sum_{i=1}^M A_i^T A_i$

$$A_i = \begin{pmatrix} a_{(i)11}, a_{(i)12} & \dots & a_{(i)1n} \\ a_{(i)21}, a_{(i)22} & & a_{(i)2n} \\ \vdots & \ddots & \vdots \\ a_{(i)m1}, a_{(i)m2} & \dots & a_{(i)mn} \end{pmatrix} = \begin{pmatrix} a_{(i)1}^T \\ a_{(i)2}^T \\ \vdots \\ a_{(i)m}^T \end{pmatrix} \tag{9}$$

And  $a_{(i)h}^T = (a_{(i)h1}, a_{(i)h2}, \dots, a_{(i)hn})$ ,  $h = 1, 2, \dots, m, i = 1, 2, \dots, M$ .

Then  $S_t = \sum_{i=1}^M A_i^T A_i = \sum_{i=1}^M \sum_{h=1}^m a_{(i)h} a_{(i)h}^T$ . Hence, the two methods are equivalent.

### 2.3 Selection of Principal Component

As 1DPCA, 2DPCA is a standard decorrelation technique and derives a set of orthogonal bases. In recent years, it has been suggested that the largest  $d$ -principal components are selected, and the corresponding eigenvectors have been selected as the bases. This would satisfy the minimum mean squared error criterion. However from pattern classification criterion, are there any contributions from the smaller principal components? According to the theory in section 2.2, and as 1DPCA we define within-class scatter matrix  $S_W$  and between-class scatter matrix  $S_b$  of image matrix as follows:

$$S_W = (1/M) \sum_{i=1}^C \sum_{j \in A_i} (A_j - \bar{A}_i)^T (A_j - \bar{A}_i) \tag{10}$$

$$S_b = (1/M) \sum_{i \in c} M_i (\bar{A}_i - \bar{A})^T (\bar{A}_i - \bar{A}) \tag{11}$$

then,

$$X_{pca_i}^T S_t X_{pca_i} = \lambda_{pca_i} = X_{pca_i}^T S_W X_{pca_i} + X_{pca_i}^T S_b X_{pca_i} \tag{12}$$

where  $X_{pca_i}^T S_W X_{pca_i}$  can be viewed as the distance of within class to which  $X_{pca_i}$  contributes, and  $X_{pca_i}^T S_b X_{pca_i}$  is the distance of between-class to which

$X_{pca_i}$  contributes. When  $\lambda_{pca_i}$  closes to zero, both  $X_{pca_i}^T S_W X_{pca_i}$  and  $X_{pca_i}^T S_b X_{pca_i}$  tend to zero. For pattern recognition, we want to select the eigenvectors that maximize the between-class distance while minimizing the within-class distance. Is the eigenvectors corresponding to the largest eigenvalues satisfying the condition? The PolyU Palmprint Database with one hundred persons and three training images/person is used to perform an experiment as an example. Fig.1 plots a graph of index  $i$  of projection vector  $X_{pca_i}$  (x-axis) against values of  $X_{pca_i}^T S_b X_{pca_i} / X_{pca_i}^T S_W X_{pca_i}$  (y-axis). The smaller the index  $i$  represents the projection vector with larger eigenvalue.

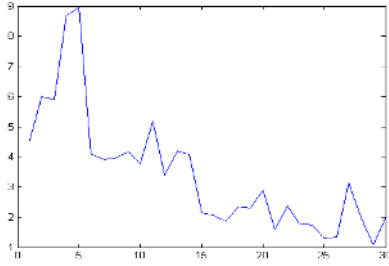


Fig. 1. Values of  $X_{pca_i}^T S_b X_{pca_i} / X_{pca_i}^T S_W X_{pca_i}$

Figure.1 shows that even  $i$  is larger than 20, there exists a value of  $X_{pca_i}^T S_b X_{pca_i} / X_{pca_i}^T S_W X_{pca_i}$  which is larger than that in the range of 15 and 20. This intuitive observation indicates that some of the smaller principal components have better balance on maximizing the between-class distance while minimizing the within-class distance than the selected large principal components. Therefore, a better strategy for selecting some smaller projection vector as the bases is required. So we define a method to select projection vector as follow:

$$X_{opt} = \arg \max(X_{pca_i}^T S_b X_{pca_i} / X_{pca_i}^T S_W X_{pca_i}) \tag{13}$$

In our method we select the principal component that have better balance on maximizing the between-class distance while minimizing the within-class distance than the large principal components.

### 2.4 Feature Extraction Method

As described in the above, the optimal projection vectors (basis vector) of 2DPCA,  $X_1, \dots, X_d$ , are used for feature extraction. For a given image sample  $A$ , let

$$Y_k = AX_k, k = 1, 2, \dots, d \tag{14}$$

Then, we obtain a family of projected feature vectors,  $Y_1, \dots, Y_d$ , which are called the principal component (vectors) of the sample image  $A$ .

Now we know that this method just reduces dimension in row direction. Obviously it is not enough, we should reduce the dimension in column direction. As described in section 2.2, 2DPCA is to do PCA on line of the image. So we should do another PCA on the column of the image. Then we can reduce dimension in row and column direction. And detail information can be obtained from [6]. Then, the last feature can be obtained by

$$Y = W_{col}^T A W_{row} \quad (15)$$

where  $W_{col}$  and  $W_{row}$  are the projection matrixes on column direction and row direction separately. It should be noted that each principal component of 2DPCA is a vector, whereas the principal component of PCA is a scalar. The principal component vectors obtained are used to form an  $d \times d$  matrix, which is called the feature matrix or feature image of the image sample A.

## 2.5 Classification Method

After a transformation by improved 2DPCA, a feature matrix is obtained for each image. Then, a nearest neighbor classifier is used for classification. Here, the distance between two arbitrary feature matrices, A and B, is defined by

$$d_F(A, B) = \left( \sum_{i=1}^m \sum_{j=1}^d (a_{ij} - b_{ij})^2 \right)^{1/2} \quad (16)$$

and it is called Frobenius distance. Suppose that the training samples are  $B_1, B_2, \dots, B_M$  (where  $M$  is the total number of training samples), and that each of these samples is assigned a given identity (class)  $\omega_k$ . Given a test sample B, if  $d(B, B_j) = \min_j (d(B, B_j))$

and  $B_j \in \omega_k$ , then the resulting decision is  $B \in \omega_k$ .

## 3 Experiments

Palmprint sampling is low cost, non-intrusive, and palmprint has a stable structural feature, making palmprint recognition the object of considerable recent research interest. Here we use the PolyU palmprint database to test the efficiency of the proposed method. The PolyU palmprint database contains 600 grayscale image of 100 different palms with six samples for each palm. Six samples from each of these palms were collected in two sessions, where the first three samples were captured in the first session and the other three in the second session. The average interval between the first and the second session was two months. In our experiments, sub-image of each original palmprint was cropped to the size of 128×128 and pre-processed by histogram equalization.



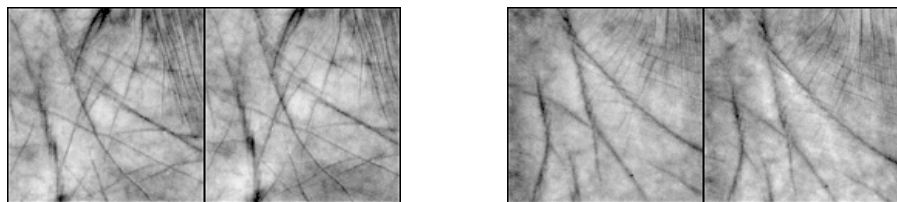


Fig. 2. Four palmprint image from three palms

Figure 2 shows four palmprint images of two palms. These images has been cropped to the size of 128×128 and pre-processed by histogram equalization. For the PolyU palmprint database, we choose the first 3 samples per individual for training, and thus use all the 300 images captured in the first session as training set and the images captured in the second session as testing set. We compare our method(I-2DPCA) with 1DPCA and 2DPCA, and the result is as follows:

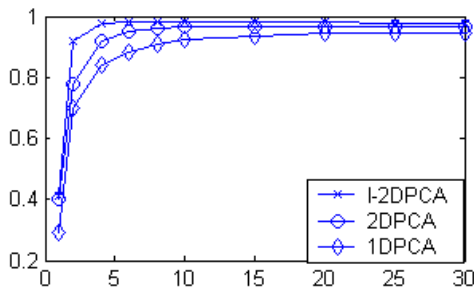


Fig. 3. Recognition rate of different method

Figure 3 shows the recognition rate obtained by different methods. From the figure 3, we can conclude that our method (I-2DPCA) is better than 2DPCA and 1DPCA.

Table1 shows the different recognition rate with different number of training samples. The recognition rate in the table is the highest under the number of training samples. And it also shows that our method (I-2DPCA) is better than the other two.

Table 1. Recognition rate of different numbers of training samples(%)

训练样本数目	1DPCA	2DPCA	I-2DPCA
1	69.6	76.7	77.1
2	86.3	90.2	92.6
3	93.6	96.3	98.2
4	94.2	97.5	98.3

## 4 Conclusion

In this paper we apply 2DPKA to palmprint recognition and proposed a method to select principal components that are better for patten recognition, and applied the improved 2DPKA to the row and column direction of the image. And the experiment result shows that our method is better for pattern recognition.

## References

- [1] Xiangqian Wu, Kuangquan Wang, Zhang. D., "A novel approach of palm-line extraction", *Proc. Third int'l Conf. Image and Graphics*. 18-20 Dec.2004. pp. 230-233
- [2] Xiang-Qian Wu, Kuan-Quan Wang, David Zhang, "Wavelet based palm print recognition", *Proc. IEEE int'l Conf. Machine Learning and Cybernetics*, vol. 3, 4-5 Nov. 2002, pp.1253 – 1257.
- [3] L. Sirovich, M. Kirby, "Low-Dimendional Procedure for Characterization of Human Faces" *J. Optical Soc. Am*, vol. 4, 1987, pp. 519-524.
- [4] M. Kirby, L. Sirovich, "Application of the KL procedure for the characterization of Human Faces" *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 12, no.1, 1990, pp. 103-108, Jan.
- [5] Wang-Meng Zuo, Kuang-Quan Wu; Zhang D; "Assembled Matrix Distance Metric for 2DPKA-Based Face and Palmprint Recognition". *Proc. Fourth IEEE int'l Conf. Machine Learning and Cybernetics*, vol. 8, 18-21. Aug. 2005, pp. 4870-4875
- [6] Wang L; Wang X; Feng J; "On Image Matrix Based Feature Extraction Algorithms", *Systems, Man and Cybern-etics, IEEE Trans.*, vol. 36, no. 1, 2006, pp. 194-197
- [7] Wangmeng Zuo, Kuangquan ang, David Zhang, "Bi-Dierectional PCA with Assembled MatrixDistance Metric", *Proc. 12rth IEEE int'l Conf.ICIP*, Vol. 2, 11-14 Sept. 2005 pp.958-961
- [8] P.N. Belhumeur, J.P. Hespanha, D.J. Kriegman, "Eigenfaces vs.Fisherfaces: recognition using class specific linear projection", *IEEE Trans. Pattern Anal. Mach. Intell.* 19(7), 1997, pp.711-720.
- [9] PolyU Palmprint database available: <http://www.comp.polyu.edu.hk/~biometrics/>

# A Combination Framework for Semantic Based Query Across Multiple Ontologies

Yinglong Ma<sup>1,2</sup>, Kehe Wu<sup>1</sup>, Beihong Jin<sup>2</sup>, and Wei Li<sup>1</sup>

<sup>1</sup> School of Computer Sciences and Technology,  
North China Electric Power University, Beijing 102206, P.R. China  
m\_y\_long@otcaix.iscas.ac.cn, epuwkh@126.com

<sup>2</sup> Technology Center of Software Engineering, Institute of Software,  
Chinese Academy of Sciences, P.O. Box 8718, Beijing 100080, P.R. China  
jbh@otcaix.iscas.ac.cn

**Abstract.** It is crucial for ontology engineers to compose these heterogeneous ontologies for effective and scalable interoperability among Web agents. In this paper, we propose a combination framework based on effective and scalable ontology interoperability for high-quality query answering. We also put forward a specific process to substantiate our combination approach. The underlying theoretical and empirical analyses are exploited for ensuring effectiveness and correctness of the approach. Experimental results from our prototype system show that high-quality query results but sacrifice some of reasoning complexity.

## 1 Introduction

It is crucial problem for ontology engineers to compose these heterogeneous ontologies for effective and effective interoperability among Web agents. In this paper, we focus on two dimensions with respect to ontology interoperability: 1) *Effectiveness*, i.e., we need to exploit high-quality methods for reducing ontological heterogeneities and providing more exact information that users want indeed. 2) *Scalability*, i.e., because these distributed ontologies probably tend to evolve, that is, their ontological contents are continually changes, methods composing these ontologies should be scalable with respect to their changes. Considering all above, we proposed a combination framework for effective and scalable ontology interoperability. Specifically speaking, we employ the combination of automatic ontology mapping (AOM) and semantic-based reasoning (SBR), and fully utilize their respective merits for efficiently performing specific tasks. This paper concentrates on some user tasks as semantic based query answering.

This paper is organized as follows: Section 1 gives a brief background about ontology interoperability. Section 2 introduces some basic terminologies. In Section 3, we specifically discuss our combination method. We propose a feasible process to substantiate our combination approach. Meanwhile, the underlying theoretical foundation and algorithm are discussed. Section 4 evaluates our combination method. Section 5 and Section 6 are the discussion and conclusion, respectively.

## 2 Preliminary

**Definition 1.** *Ontology  $\mathcal{O}$  can be defined as a seven tuple:*

$$\mathcal{O} := (\mathcal{C}, \mathcal{H}_C, \mathcal{R}, \mathcal{H}_R, \mathcal{P}, \mathcal{A}, \mathcal{D})$$

where  $\mathcal{C}$  is the set of concept  $C$ .  $\mathcal{H}_C \subseteq \mathcal{C} \times \mathcal{C}$  denotes subsumption relation between concepts.  $\mathcal{R} \subseteq \mathcal{C} \times \mathcal{C}$  denotes the set of binary property relation between concepts, whereas  $\mathcal{H}_R \subseteq \mathcal{R} \times \mathcal{R}$  denotes hierarchy between properties.  $\mathcal{P} \subseteq \mathcal{D} \times \mathcal{DT}$  is a set of data properties for concept, where  $\mathcal{DT} \subseteq \mathcal{D}$  is a datatype set.  $\mathcal{A}$  is a set of axioms expressed in a logical language and can be used to infer knowledge from existing one.  $\mathcal{D}$  denotes a domain set containing all instances of concepts  $C$ .

Ontology Mapping between ontology  $\mathcal{O}_1$  and  $\mathcal{O}_2$  can be regarded as follows: some entities in  $\mathcal{O}_1$  is mapped onto at most one entity in  $\mathcal{O}_2$ , vice versa.

**Definition 2.** *Let  $\Sigma_{\mathcal{O}}$  be vocabulary of ontology  $\mathcal{O}$ . We use  $\varepsilon$  to refer to a null entity. Ontology mapping can be defined by a mapping function  $\rho : \Sigma_{\mathcal{O}_s} \rightarrow \Sigma_{\mathcal{O}_t}$  such that:*

$$\forall s \in \Sigma_{\mathcal{O}_s} (\exists t \in \Sigma_{\mathcal{O}_t} : \rho(s) = t \text{ or } \rho(s) = \varepsilon)$$

where  $\Sigma_{\mathcal{O}_s}$  and  $\Sigma_{\mathcal{O}_t}$  are the source ontology and the target ontology of mapping  $\rho$ . Sometimes, we can use  $\rho_{\mathcal{O}_1, \mathcal{O}_2}$  denotes the mapping function from  $\mathcal{O}_1$  to  $\mathcal{O}_2$ .

Semantic reasoning over ontology representations can help us to find more implicit semantic information.

**Definition 3.** *The semantic representation of  $\mathcal{O} := (\mathcal{C}, \mathcal{H}_C, \mathcal{R}, \mathcal{H}_R, \mathcal{P}, \mathcal{A}, \mathcal{D})$  can be defined based on an interpretation  $\mathcal{I} = (\mathcal{D}, \bullet^{\mathcal{I}})$ , where  $\mathcal{D}$  is the non-empty domain set, and  $\bullet^{\mathcal{I}}$  is the interpretation function.  $\bullet^{\mathcal{I}}$  maps concept  $C \in \mathcal{C}$  into a set  $C^{\mathcal{I}} \subseteq \mathcal{D}$ , maps  $R \in \mathcal{R}$  into  $R^{\mathcal{I}} \subseteq C^{\mathcal{I}} \times C^{\mathcal{I}}$ , maps  $Hc \in \mathcal{H}_C$  into  $Hc^{\mathcal{I}} \subseteq C^{\mathcal{I}} \times C^{\mathcal{I}}$ , and maps  $Hr \in \mathcal{H}_R$  into  $Hr^{\mathcal{I}} \subseteq \mathcal{R}^{\mathcal{I}} \times \mathcal{R}^{\mathcal{I}}$ .*

We say that  $\mathcal{I}$  is an interpretation (model) of  $\mathcal{A}$  iff for every axiom  $A \in \mathcal{A}$ ,  $\mathcal{I} \models A$ .  $\mathcal{A} \models A$  iff for every interpretation  $\mathcal{I}$  of  $\mathcal{A}$  such that  $\mathcal{I} \models A$ . A concept  $C$  is satisfiable iff  $C^{\mathcal{I}} \neq \emptyset$ ;  $\mathcal{A}$  is consistent iff there exists a model  $\mathcal{I}$  of  $\mathcal{A}$ ;  $\mathcal{O}$  is consistent iff there exists a model  $\mathcal{I}$  of  $\mathcal{O}$ . As for an element  $\alpha$  in  $\mathcal{O}$ ,  $\mathcal{O} \models \alpha$  iff for all interpretation  $\mathcal{I}$  of  $\mathcal{O}$ ,  $\mathcal{I} \models \alpha$ . Through axiom set  $\mathcal{A}$ , we can perform logical inference for and find out implicit information from ontology.

## 3 A Combination Framework

This section specifically introduces our combination method and discusses some important aspects such as canonical process, inconsistency elimination and its underlying terminology approximation theory and algorithms.

**1. Automatical ontology mapping.** We borrow the methods from [6]. We first compare features from different ontologies. These features include OWL labels such as `rdfs:Class`, `rdfs:Property`, `rdfs:subClassOf`, `owl:sameAS`,

`rdfs:subPropertyOf`, `rdf:type`, `owl:disjointWith`. Some domain specific features are also be taken into account. As for documents suffixed with `.pdf`, `.doc`, `.txt`, they can be regarded as specific course objects that are instances of concepts. Then using aggregation of multiple strategies of similarity comparison, we compare these features and measure their aggregated similarity. The process of automatical ontology mapping will probably take place in an iterative way. At last, we will obtain these similarity of different entities.

## 2. Elimination of Inconsistent Ontology

**Definition 4.** We use  $A \sqsubseteq B$  for denoting the axiom that concept  $A$  is a subclass of concept  $B$ . Let  $\alpha = A \sqsubseteq B$  and  $\beta = C \sqsubseteq D$  be two axioms in  $\Sigma$ . If  $A^{\mathcal{I}} \cap C^{\mathcal{I}} \neq \emptyset$  but  $B^{\mathcal{I}} \cap D^{\mathcal{I}} = \emptyset$ , then axioms  $A \sqsubseteq B$  and  $B \sqsubseteq C$  are connected and denoted as  $connected(\alpha, \beta)$ .

If we find there are two axioms are connected, we must eliminate ontology inconsistencies they probably bring about. We propose an adapted algorithm based on Haase’s methods [11] for handling inconsistencies of changing ontologies after accomplishing automatical mapping. The revised algorithm is used for obtaining a maximal consistent subontology after we automatically generate all ontology mappings.

Let  $\Sigma = \{\Sigma_{\mathcal{O}}\}$  be a vocabulary based on all possible mapping ontologies  $\mathcal{O}$ ,  $\mathcal{M}$  is concept subsumption relations based on all possible ontology mappings. This revised algorithm is listed as follows.

---

### Algorithm. *Find-Maximal-Consistent-SubOntology*( $\Sigma, \mathcal{M}$ )

---

```

Require :  $\Sigma$ , a vocabulary about all possible ontologies
Require :  $\mathcal{M}$ , a mapping axiom set of ontology mappings
begin
forall  $\alpha \in \mathcal{M}$  do
   $\Omega := \{\Sigma \cup \alpha\}$ 
  repeat
     $\Omega' = \emptyset$ 
    forall  $\Sigma' \in \Omega$  do
      forall  $\beta_1 \in \Sigma' \setminus \{\alpha\}$  do
        if there is a  $\beta_2 \in (\{\alpha\} \cup (\Sigma \setminus \Sigma'))$  such that  $connected(\beta_1, \beta_2)$  then
           $\Omega' := \Omega' \cup \{\Sigma' \setminus \{\beta_1\}\}$ 
        endif
      endfor
    endfor
  until there exists an  $\Omega' \in \Omega$  such that  $\Omega'$  is consistent
   $\Sigma := \Omega'$ 
endfor
end

```

---

**3. Reasoning implicit Knowledge.** This section will discuss how implicit knowledge in the DAG-like ontologies can be mining out. In Step 1, we have found the semantic mappings of different ontologies using automatical ontology mapping. For example, `Programming_Language` is a subclass of both `Software_Engineering` and `Language`, both `Structural_Programming_Language` and `OOPL` are subclasses of `Programming_Language`. Through simple semantic reasoning, we will find `OOPL` and `Structural_Programming_Language` also are subclasses of the class `Software_Engineering`. These knowledge is hidden in

Ontology. What we need to do is just to perform semantic reasoning based on the consistent ontology and explicitly add those new knowledge into  $\Sigma$ .

**4. Query Approximation.** Query approximation can be used for user query answering and further achieve better semantic coordination for Web agents. Here we adopt terminology rewriting in [2] for approximate query answering. Its underlying theoretical foundation is simple and its correctness can be ensured in [2]. A Web agent called EPU wants to query `ProgrammingLanguage $\wedge$  $\neg$ SoftwareEngineering`. Because of terminological semantic heterogeneities, they must coordinate with each other. Then EPU agent delegates its query to Web agent called IOS. The terminology `ProgrammingLanguage` should be rewritten as terminology `Language $\wedge$  $\neg$ SoftwareEngineering`. Furthermore, a specific algorithm is provided for terminology rewriting in [2].

**5. Semantic based query answering.** From the viewpoint of semantic query theory, a semantic query  $\alpha$  answers can be obtained by checking satisfiability of the query [9]. The consequent set can be formally represented as  $\{\alpha : \mathcal{O} \models \alpha\}$ . For example, considering a query  $C \wedge D$ , the answers are the set of all objects  $o$  such that for all interpretation  $\mathcal{I}$  of  $\mathcal{O}$ ,  $o \in C^{\mathcal{I}} \wedge o \in D^{\mathcal{I}}$ , i.e., all objects  $o$  make both  $C$  and  $D$  satisfiable. From the view of system implementation, semantic based query is performed by querying a set consisting of some triples of the form (subject, predicate, object), whereas the set just represents all information of the queried ontology. For example, we assume that there is a very simple ontology whose namespace is Home and which is represented as a set  $\mathcal{O} = \{(\text{Home:Person}, \text{rdf:type}, \text{rdfs:Class}), (\text{Home:Father}, \text{rdf:type}, \text{rdfs:Class}), (\text{Home:Father}, \text{rdfs:subClassOf}, \text{Home:Person}), (\text{Home:Joe}, \text{rdf:type}, \text{Home:Person}), (\text{Home:John}, \text{rdf:type}, \text{Home:Father})\}$ . Then, we should tackle its inconsistency (if any) and reason this ontology and mine out implicit information. In this ontology, the explicit information `(Home:John, rdf:type, Home:Person)` will be mined out and added into  $\mathcal{O}$ . If we want to query `Person`, we just find out all instances (objects) belonging to `Person`. The answer results should be Joe and John.

## 4 Empirical Evaluation and Discussion

### 4.1 Experiment Evaluation

**1. Evaluation Index.** Standard information retrieval metrics is used to evaluate our combination method in this section.

$$\text{Precision(P)} = \frac{|\text{correct\_discoverd\_objects}|}{|\text{discovered\_objects}|}; \text{ Recall(R)} = \frac{|\text{correct\_discoverd\_objects}|}{|\text{correct\_objects}|}$$

where *correct\_discoverd\_objects* is disjunction of the set consisting of correct objects and the set consisting of discovered objects. *correct\_objects* should be disjunction of sets of correct objects from the multiple data sources.

**2. Data Sets.** Here we make evaluation on three data sets.

i) **EPU-IOS 1.** The first data set includes EPU Course ontology and IOS Course ontology, respectively. They are created according to contents of independent websites about Department of Computer Sciences of EPU and Institute of Software. EPU ontology includes 59 concepts and 143 individuals. IOS course ontology includes 63 concepts and 157 individuals.

ii) **EPU-IOS 2.** The second data set also covers the courses from IOS and EPU. But they are described by students from IOS and EPU. They probably changed some course labels and describe their courses in different ontology structures. EPU ontology includes 43 concept entities and 143 individuals. IOS course ontology includes 51 concept entities and 157 individuals.

iii) **ShowPlace.** The third data set includes two ontologies separately describing some showplaces from China. All placename terminologies of the two ontologies are randomly and objectively collected from Websites Baidu<sup>1</sup> and Netease<sup>2</sup>. Both ontologies have an extent of about 150 entities. Some relative pictures about these places are also collected as instances of ontologies.

**3. Evaluation Strategies.** Furthermore, we evaluate the precision and recall in the following different situations:

- 1) MS, i.e., manual ontology mapping + semantic based query
- 2) QK, i.e., quick ontology mapping + keyword based query
- 3) QS, i.e., quick ontology mapping + semantic based query
- 4) QA, i.e., quick ontology mapping + approximate semantic based query

We first give some queries based on the three data sets, then compute their average precision and average recall with respect to these queries. Here, keyword based query means that we don't use semantic reasoning but directly match query strings with ontology terminology strings. Through the comparisons among their precision and recall in the different situations, we can get some useful information for instructing our future work. The evaluation results are shown in Figure 1 to 2.

## 4.2 Empirical Results and Discussion

All of the facts show that our combination method is more effective and scalable than only using one of these methods such as quick mapping method, semantic based reasoning, keyword based query.

QOM claims that their theoretical complexity is  $O(n \log(n))$  [6]. As for the inconsistency elimination algorithm, its theoretical complexity is  $O(n^4)$ . Theoretical complexity of semantic reasoning is variable in accord with contents of ontology presentation. If there are no property hierarchies in ontologies, then its complexity should be about  $O(n^2)$ . Otherwise, its reasoning complexity will be polynomial [10]. Theoretically speaking, it will lead to more complexity for more

<sup>1</sup> www.baidu.com

<sup>2</sup> www.netease.com

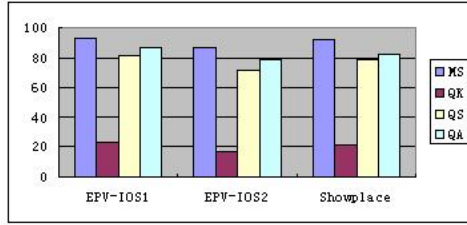


Fig. 1. Comparison of query precision

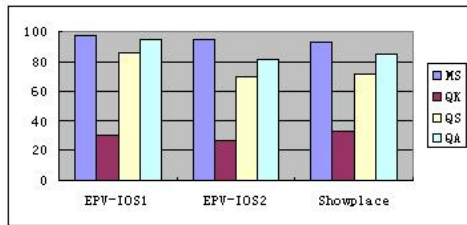


Fig. 2. Comparison of query recall

expressive ontology representation such as property hierarchies. For simplification, this paper doesn't consider property hierarchies. A query is translated into its approximate one in  $O(n)$  (page.13,[2]) complexity.

So we will find that theoretical complexity of our method is the sum of the followings:  $O(n \log(n)) + O(n^4) + O(n) + O(n^g)$ . Specifically speaking, theoretical complexity of our method is:

$$\begin{aligned}
 &O(n^g), \text{ if } g > 4 \\
 &O(n^4), \text{ otherwise}
 \end{aligned}$$

where  $g$  is the number with highest exponent in the polynomial. To certain extents, computation complexity based on our method increases because of the need for semantic reasoning.

## 5 Related Work

There are some research works that concentrate on ontology inteoperability. Semantic mapping between ontologies is considered as a good approach for establishing ontology interoperability [1,2,3,4]. However, their quality evaluations to ontology interoperability greatly rely on prior mappings that are established against human judgements or the commonsense. They didn't provide solutions to efficiently and automatically establish or discovery semantic mappings between partial ontologies. There are also some researchers that concentrate on scalability of mappings [5,6,7]. Their work attempted to automatically and efficiently establish mappings between ontologies by borrowing some theory and technologies such as similarity measurements [8]. Unfortunately, they seldom rely



on underlying logic and semantic based reasoning of ontology representation. So it is difficult to ensure more effective results for given semantic queries. In brief, these methods have their respective merits and limitations.

In contrast, our method attempts to make full use of their respective merits and avoid or reduce their limitations. In this paper, we propose a combination for automatical ontology mapping and high quality approximate query answering. Considering effectiveness and scalability of distributed ontologies, we employ the combination of automatic ontology mapping (AOM) and semantic-based reasoning (SBR), and fully utilize their respective virtues for efficiently performing specific user tasks such as query answering. In fact, the experimental results show that this method is very effective and can obtain high-quality query results.

## 6 Conclusion

We propose a combination for effective and scalable ontology interoperability, and empirical results show that it is effective and scalable. The contribution of this paper includes: **1)** We first combine automatical ontology mapping with approximate query answering for more effective and scalable ontology interoperability. **2)** In our opinion, there probably are subsumption inconsistencies in ontologies that are automatically constructed mappings. So we propose a revised algorithm for eliminating ontology inconsistencies. **3)** We give a canonical process for the combination framework. The empirical results show that it is effective and scalable for ontology interoperability. Future work includes: 1) applying this combination method to more larger systems 2) exploiting more intelligent and efficient algorithms for reducing reasoning complexity.

## Acknowledgements

This work is supported by National Natural Science Fund (No.60573013). We also thank anonymous referees for their valuable comments.

## References

1. Agrawal, R. and Srikant, R.: On Integrating Catalogs. In: Proceedings of WWW2004, 2004
2. Stuckenschmidt, H.: Exploiting Partially Shared Ontologies for Multi-Agent Communication. In: Proceedings of CIA'02, 2002
3. Noy, N., et al.: The Prompt Suite: Interactive Tools for Ontology Mergin and Mapping. In: *International Journal of Human-Computer Studies*, **59(6)**, (2003) 983–1024
4. Akahani, J., et al.: Approximate query reformulation for ontology integration. In: Proceedings of ISWC2003, 2003
5. Doan, A., et al.: Learning to Map between Ontologies on the Semantic Web. In: Proceedings of WWW2002, 2002

6. Ehrig, M and Staab, S.: QOM - Quick Ontology Mapping. In: Proceedings of ISWC2004, 2004
7. Tang J., et al.: Using Bayesian Decision for Ontology Mapping. In *Journal of Data Semantics*, 2006
8. Maedche, A., et al.: Mafra - A Mapping Framework for Distributed Ontologies. In: Proceedings of EKAW2002, 2002
9. Horrocks, I., et al.: Querying the Semantic Web: A Formal Approach. In: Proceedings of ISWC2002, **2342**, (2002)177–191
10. Horrocks, I., et al.: Practical Reasoning for very expressive description logics. In: *Logic Journal of IGPL*, **8(3)**, (2000)239–264
11. Haase, P., et al.: A Framework for Handling Inconsistency in Changing Ontologies. In: Proceedings of ISWC2005, 2005

# Adaptive Mechanisms of Organizational Structures in Multi-agent Systems

Wang Zheng-guang, Liang Xiao-hui, and Zhao Qin-ping

Department of Computer Science and Engineering, Beihang University, P.R. China, 100083  
{wangzg, lxxh, zhaoqp}@vr1ab.buaa.edu.cn

**Abstract.** Organizational adaptation is one of the key issues of organization and reorganization theories. As the result of environmental changes, organizations can change their internal structures or agents in the organizations can adjust their behaviors to achieve organizational objectives. In this paper, we focus on the characteristics of organizational structures and also analyze the notion of organizational stability and its changes resulted from the constraints of roles and their relations in the organizational structure. In the end, we give a formal specification of the adaptive mechanisms for organizational structures in details to meet dynamic environment requirements.

## 1 Introduction

Multi-agent methodologies have commonly been adopted to solve complex and dynamic problems. In order to coordinate individual activities of agents during the process of problem solving, organizations are well recognized as an efficient way to decrease conflicts between agents' activities. However, since existing in dynamic and open environments, agent organizations have to cope with environmental changes, i.e. agents leaving the organizations possibly results in the failures of achieving organizational objectives, which may lead to structural transition of organizations[10] or emergent changes of agents behaviors(e.g. behavioral changes[13]). Analogous to human organizations, agent organizations also conform to specific organizational rules, pattern and structures [15]. Especially, organizational structures describe how agents in the organizations should coordinate their activities to achieve organizational objectives efficiently. We aim at how agent organizations alter main elements of their structures to satisfy the dynamic requirements of environmental changes. The issue presented here is motivated from the following considerations.

First of all organizations are well recognized for existing on certain objectives, which are either pre-defined by designers or formed temporarily by emergently cooperative agents. Organizational objectives are often of structural characteristics, i.e. an objective consists of several sub-objectives, which can constrain the relationships of agents in different types of organizational structures (e.g. market, network and hierarchy organizations [3]), that is, agents with a shared objective should choose the corresponding sub-objectives. Organizational structures image the structural properties of organizational objectives in different application contexts.

Secondly, organizational stability reflects the survival capabilities of an agent organization showing whether organizational objectives are reached in specific

scenarios. Organizational stability changes indicate that changes appear from the structures and compositions of organizations. We will analyze the notion of organizational stability and its changes based on organizational structure.

Finally, in order to improve self-adaptation of multi-agent systems, previous researches concentrate on dynamic reorganization [7], [13], organizational self-design [1], [11], organizational transition [10], monitoring and regulating emergent organizational structures to avoid undesirable behaviors [14]. Organizational structures are viewed as one of the key factors in studying adaptive mechanisms of organizations. We formulate the adaptive mechanisms of organizations based on the transformation of organizational structures.

In this paper, we discuss the adaptive mechanisms of organizational structures that improve robustness and survival capabilities of multi-agents systems. In section 2, the features of organizational structures are introduced. Organizational stability based on structural properties of organizations is specified and the formal representation of adaptive mechanisms is also discussed in section 3. Finally, some conclusions and future researches are given in section 4.

## 2 Features of Organizational Structures

Previous work on agent organizations focuses on central concepts: namely that of agents, groups, roles and their interrelations (e.g. organization models such as AGR, MOISE+, ISLANDER and etc. in [9]). The existent methodologies of analyzing organizational structures are distinguished by agent-centered and organization-centered approaches. The former impose its emphasis on agent autonomy and individual dependences, by which agents can fulfill their own objectives, and moreover, emergent cooperative behaviors may appear among agents when it is necessary for agents to perform shared tasks together (e.g. agent dependence graph in [8], agents with norms [4]). Emergent social structures are built on agent dependence graphs. The latter concentrates on main elements of organizational structures, i.e. roles, groups, and social structures consisting of roles. Roles and their interrelations are the central notions based on the structural dimension of organization-centered approaches. D. Grossi et al [2] describe three dimensions of the relations of roles in organizational structure including power, coordination and control, and gives a formal specification of these kinds of relations in organizational structure and the effect of them on actions of agents in the organization. From the view of organization, organizational objectives and their interrelations reflect the properties and requirements of organizational structures.

### 2.1 Roles and Dependences

In existent literature, typical relations between roles mainly consider dependences, power, authorization and right [1], [2], [3]. Role dependences shape the solid social structure constraining roles in an organization, and imply other relations such as power and authorization [1]. A role should relate to certain objectives which are usually of an inherent hierarchy. We give a specification of role objective and role dependences according to role objectives as follows.

**Definition 1 (Role Objective)**

A role objective is a given organizational objective. The set of objectives of a role  $r$  is expressed by  $G^r$ , which can not be empty. For two roles  $r_1$  and  $r_2$ ,  $r_1$  equals to  $r_2$  if, and only if,  $G^{r_1}$  is the same to  $G^{r_2}$ .

For an objective  $g$  of a role  $r$ , the set of sub-objectives of  $g$  is expressed by  $\vartheta_g^r$ .

Role dependences describe the distribution of objectives among different roles, that is, the hierarchical constraints of objectives also influence dependences between roles as shown in Definition 2.

**Definition 2 (Constraints of Role Dependences)**

Given a set  $\Pi^{role}$  of roles, there is a partial order,  $\leq_{role}$ , over two roles with dependence. For  $\forall r_1, r_2 \in \Pi^{role}$ , if  $r_2 \leq_{role} r_1$ , that is to say, in the given organizational structure,  $G^{r_2}$  is a set or sub-set of sub-objectives of some objective in  $G^{r_1}$ .

## 2.2 The Graph Based Organizational Structure

Based on the partial orders on roles, role dependences constitute the organizational structure expressed by a labeled, directed acyclic graph named Rolegraph. The definition of Rolegraph and its properties are given here.

**Definition 3 (Rolegraph of Organizational Structure)**

Given a set  $\Pi^{role}$  of roles consisting of identifiers (e.g. generic ASCII strings) and a partial order  $\leq_{role}$  over  $\Pi^{role}$ , a rolegraph  $RG$  of organizational structure is a tuple:

$$RG = \langle V, E, source, target, label \rangle \quad (1)$$

In equation (1),  $V$  is a finite set of entities representing roles.  $E$  is a finite set of edges,  $E \subseteq V \times V$ . The mappings  $source$  and  $target$  assign a source and a target node to each edge denoted by  $source, target : E \mapsto V$  respectively and the mappings  $label$  assigns an identifier to each node in  $V$  or edge in  $E$  denoted by the equation (2).

$${}^1 label : V \cup E \mapsto \Pi^{role} \times T \quad (2)$$

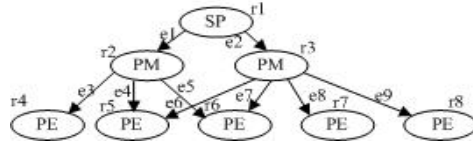
$T$  is any set of desirable identifiers (e.g. generic ASCII strings). An example of the rolegraph of organizational structure is shown in Fig.1.

**Definition 4 (Subgraph and Occurrence Morphism)**

A graph  $L$  is a subgraph of the rolegraph  $RG$  denoted by  $L \subseteq RG$  if, and only if, the node and edge sets of  $L$  are subsets of the corresponding sets of  $RG$ , and the mappings, i.e. source, target and label, coincide with the corresponding mappings of  $RG$ .

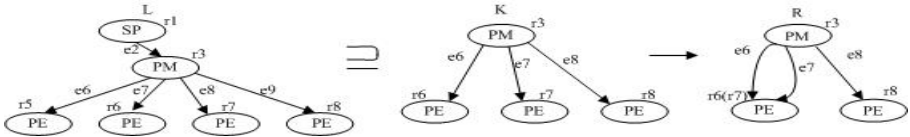
---

<sup>1</sup>  $label'$  and  $label''$  respectively the first and second components of  $label$ , for example,  $label'(V) \subseteq \Pi^{role}$  and  $label''(E) \subseteq T$ .



**Fig. 1.** An exemplar diagram of the rolegraph in an agent organization. SP, PM and PE are three roles Supervisor, Product manager and Product engineer in a manufacturing organization.

We name that the graph  $L$  has an occurrence in the rolegraph  $RG$  denoted by  $L \rightarrow RG$  if, and only if, there is a mapping that maps respectively the nodes and edges of  $L$  to the nodes and edges of  $RG$  with no changes of the mappings of sources, targets and labels. Fig.2 depicts the subgraph and occurrence of a rolegraph.



**Fig. 2.** An example of subgraph and occurrence of a rolegraph, where  $K \subseteq L$  and  $K \rightarrow R$

### 3 Specifying Organizational Stability

Organizational stability addresses the issues of whether and how to achieve organizational objectives. For the view of structural dimension, roles and their relations in organizational structures decide the capabilities in maintaining organizational stability. We focus on organizational stability based on roles and their interrelations, i.e. dependences between them with partial orders.

#### 3.1 Reasons for Organizational Stability Change

Organizational stability can be influenced by main elements of organizational structures (see e.g. behavioral and structural change in [3]), whose changes will result in the changes of organizational stability. In general, the changes of organizational structure may be distinguished by two aspects: macro and micro. The former indicates the ultimate changes of organization, i.e. redesign of organization structure and change of given organizational global objectives, and the latter reflects some transient changes of an organization, i.e. agents joining or departing from the organization and their behavior adaptation to the organization specification. Organization stability is one of the key issues of the macro-aspect, which concerns the modification of one or more structural elements of an organizational structure, i.e. change of role objectives and dependences between roles. To assure organization stability, an organization should provide some adaptive mechanisms of modifying main elements of organizational structures, but not change the organizational global objective.

### 3.2 Formal Representation of Adaptive Mechanisms

If the organizational stability changes, the role dependence graph of an organization will change from one to another, which can be described by graph transformation. Performing the transformation of the role dependence graphs by graph transformation means to transform them according to certain transformation rules, and then the process of the transformation between two dependence graphs may be expressed by a composition of transform rules. Adaptive mechanisms of organizational structures contain how to redesign and select a new role dependence graph since some role in the source graph fails to achieve its objectives. For example, Fig.3 shows two role dependence graphs, and the left side  $L$  describes a role  $r3$  failing to manage two roles  $r7$  and  $r8$ , which results in the transformation of dependence graphs from  $L$  to the right side  $R$ .

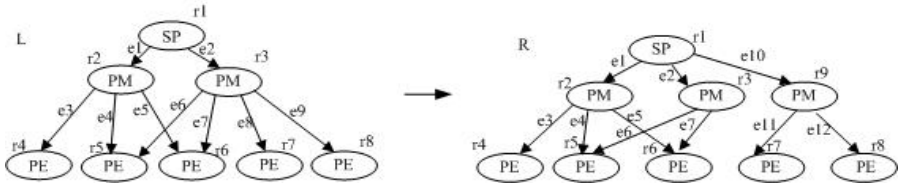


Fig. 3. Transformation between the role dependence graphs

We give a formal representation of adaptive mechanisms based graph transformation here.

#### Definition 5 (Transformation Rules)

Given two dependence graph  $L, R \in \Theta_{role}$ , a graph transformation rule  $p$  is expressed by a morphism over  $L$  and  $R$ ,  $p: L \mapsto R$ . A graph transformation rule reflects how changes of nodes and edges in given dependence graphs take place, i.e. insert or delete a node or an edge.

#### Definition 6 (Composition of Rules and Transformation)

Given a set  $\Pi_p$  of graph rules, for  $s, f \in \Pi_p$ , the composition  $s \circ f$  of functions  $f: L \mapsto T$  and  $s: T \mapsto R$  is defined by  $(s \circ f)(L) = s(f(L))$ . The transformation of two dependence graphs, i.e.  $L$  and  $R$  shown in Fig.3, denoted by  $L \Rightarrow^c R$ , consists of a composition  $c_p$  of graph rules, where  $c_p \equiv (s \circ f): L \mapsto R$ .

Based on the notions of graph transformation and rules, the specification of an adaptive mechanism is defined as follows.

#### Definition 7 (Definition of Adaptive Mechanisms)

An adaptive mechanism  $AM$  is a tuple:

$$\langle am - premiss, G_{src}, \Theta_{role}, \Pi_p \rangle$$

Where  $am - premis$  is a set of conditions indicating organizational stability changes.  $G_{src}$  is a role dependence graph in a given situation.  $\Theta_{role}$  is a set of role dependence graphs according to organizational objectives.  $\Pi_p$  is a set of graph transformation rules.

Adaptive mechanisms of organizational structures can be either designated by designers or obtained through learning during the system runtime. In order to discover the changes of organizational stability and enforce the transformation of role dependence graphs, an organization should provide special components for monitoring and controlling the adaptive process.

## 4 Conclusions

We presented a formal specification of adaptive mechanisms of organizational structures for environmental changes, which shows that organizational stability can vary based on the specific relations in organizational structures, i.e. the graph of role dependences, which are determined by organizational objectives in application contexts. According to organizational stability changes, we specified a formal representation of adaptive mechanisms to adjust structural elements and their relations of organizations to dynamic environments. Future works will concentrate on assessment of organizational stability and evolution mechanisms of organizations. Qualitative and quantitative simulation experiments on reorganization will be also exploited to complement the analytic methodologies of researches on organizations.

## References

1. B. Horling, B. Benyo, and V. Lesser: Using Self-Diagnosis to Adapt Organizational Structures. Computer Science Technical Report TR-99-64, University of Massachusetts at Amherst (1999)
2. D. Grossi, F. Dignum, L. Royackers, M. Dastani: Foundations of Organizational Structures in Multi-Agent Systems. Fourth International Conference on Autonomous Agents and Multiagent Systems, Utrecht, ACM Press (2005) 690--697.
3. Dignum, V., Dignum, F.: Structures for agent organizations. International Conference on Integration of Knowledge Intensive Multi-Agent Systems (2005) 215 – 220
4. F. Dignum: Autonomous agents with norms. Artificial Intelligence and Law 7, (1999) 69-79
5. G. Boella, Luigi Sauro and L. van der Torre: Power and Dependence Relations in Groups of Agents. Proceedings of IAT 04, IEEE (2004)
6. G. Boella and L. van der Torre: Role-based Rights in Artificial Social Systems. Proceedings of IAT05, IEEE (2005)
7. Glaser, Norbert and Morignot, Philippe: The Reorganization of Societies of Autonomous Agents. In Modelling Autonomous Agents in Multi-Agent Worlds (MAAMAW). (Ronneby, Sweden). Lecture Notes in Artificial Intelligence, Vol. 1237, Springer Verlag (1997)98-111



8. Jaime Simão Sichman and Rosaria Conte: Multi-Agent Dependence by Dependence Graphs. In Proc. 1st International Joint Conference on Autonomous Agents and Multi-Agent Systems (AAMAS'02), Bologna, Italy (2002) 483-492
9. Luciano R. Coutinho, Jaime S. Sichman, Olivier Boissier: Modeling Organization in MAS: A Comparison of Models. First Workshop on Software Engineering for Agent-oriented Systems (2005)
10. Matson, E., DeLoach, S.: Formal transition in agent organizations. International Conference on Integration of Knowledge Intensive Multi-Agent Systems (2005)235 – 240
11. So, Y. and Durfee, E. H.: An organizational self-design model for organizational change. In AAAI-93 Workshop on AI and Theories of Groups and Organizations: Conceptual and Empirical Research, Washington, D.C. (1993)8-15
12. Norman, T.J., Sierra, C., Jennings, N.R.: Rights and commitment in multi-agent agreements. International Conference on Multi Agent Systems (1998) 222 – 229
13. Virginia Dignum, Liz Sonenberg, Frank Dignum: Towards Dynamic Reorganization of Agent Societies, Proceedings of Workshop on Coordination in Emergent Agent Societies at ECAI 2004, Valencia, Spain
14. Zahia Guessoum, Mikal Ziane, Nora Faci: Monitoring and Organizational-Level Adaptation of Multi-Agent Systems. Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems (2004) 514-521
15. Zambonelli F, Jennings NR, Wooldridge M: Organizational abstractions for the analysis and design of multi-agent systems. Proceedings of the 1st International Workshop on Agent-Oriented Software Engineering. Limerick (2000) 127-141

# An Agent-Based Services Composition Framework for Ubiquitous Media

Hui Wang, Yuhui Zhao, Deguo Yang, Cuirong Wang, and Yuan Gao

School of Information Science and Engineering, Northeastern University, 110004  
Shenyang, China  
wanghui@mail.neuq.edu.cn

**Abstract.** Ubiquitous media aims to provide media services anytime and anywhere. Software agents are one of the building blocks of ambient intelligence and pervasive computing. In this paper, we use a structured overlay network as the decentralized service repository system for improved efficiency. We present an agent-based services composition framework for ubiquitous media, through a combination of the respective merits of agent and service-oriented to enables dynamic and efficient delivery of media composite services. We advocate that a service-centric context promotes media applications that allow service adaptability, deal with service availability, and support on-the-fly service composition. Finally, we present an example of our system in the context of streaming video playback involving a series of transcoding services and a mobile client.

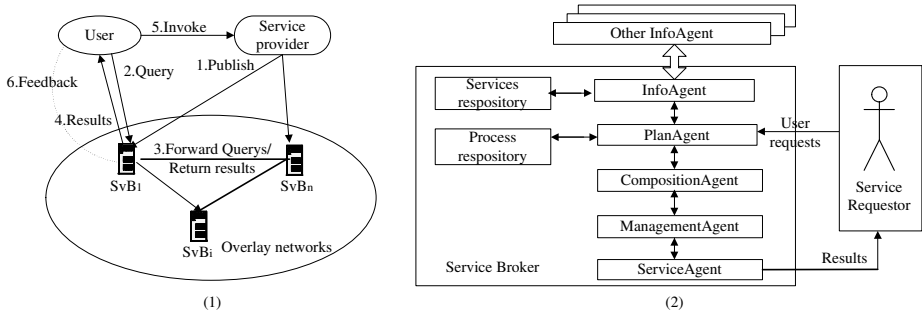
## 1 Introduction

Digital multimedia is permeating and enriching every aspect of our life. Much recent networking research has been based on the assumption that a large number of heterogeneous and likely mobile network-enabled devices may be used to access both static and streaming media content over the Internet. This trend strongly calls for a modular design approach, which is the Service-Oriented Architecture (SOA)[1]. Because of the complexity of media service composition, our proposal is to investigate a multi-agent System as a complement to a service-oriented approach to enable multimedia application for wireless clients.

There are several articles, which consider symbolic service composition [2,3]. Some essential issues in decentralized Web service provision have been considered by [4]. Towards incorporating Web Service into agents has been the subject of many research projects [5,6]. Based on these, we develop a decentralized discovery approach to achieve the high scalability and we use a structured overlay network as the decentralized service repository system for improved efficiency. We propose an agent-based dynamic services composition model, which have five types of agents. These agents cooperate with each other to implement service composition tasks. It is a service-centric context-based method, which promotes media applications in the follows: 1) allowing service adaptability, 2) dealing with service availability, and 3) supporting on-the-fly service composition.

## 2 The System Architecture

The service discovery model we use in this paper is a decentralized service repository system with a structured overlay network topology. Figure 1(1) shows the conceptual model of our distributed service discovery framework.



**Fig. 1.** (1) Service Discovery Framework (2)Service Broker internal Structure

(1) Providers publish their services with embedded QoS information to service broker nodes (SvBs) in overlay networks. (2) Users query for services with certain functionalities and required QoS levels using any SvBs as their access point. (3) The SvBs take care of routing the request to the peer(s) that can answer it. (4) The results will be returned to the user. (5) This user may invoke one of the found services. (6) Users can express feedbacks on the QoS they could obtain from a service to the SvBs managing that service.

A service broker’s internal architecture is shown in Figure 1(2). It has five types of agents that are devised namely, InfoAgent, PlanAgent, CompositionAgent, ManagementAgent and ServiceAgent to conduct service information collection, service plan generation, service composition, service management and service instance execution, respectively. Besides the five types of agents, two types of repository need to maintain. They are service repository (SR) and plan repository (PR). SR stores Web services information that is registered by service providers. PR stores process plans that are created and maintained by PlanAgent for user’s requests.

We consider a Web service as a component that is instantiated each time when it participates in a new composition. We also consider three types of services, namely, composite service, component service, and service instance. Composite services associated with CompositionAgents, component services associated with ManagementAgents and service instances associated with ServiceAgents. To enhance systems with context-aware capabilities, we define three types of context. They are CS-context, WS-context, IS-context, respectively. Each service is attached to a specific context. Composite service is attached to CS-context, component service is attached to WS-context and service instance is attached to IS-context.

### 3 The Agent-Based Services Composition Model

The following five types of collaborating agents are involved in the process of service composition.

#### 3.1 The Agent Model

**InfoAgent:** The InfoAgent keeps a known set of web services candidates for user invocation in Service Repository. All service providers publish expected values about their services. These numbers will be refined by InfoAgents in their SR according to the actual values received from user feedbacks to reflect more accurate values. A Web service may have many records in SR depending on how many operations and service levels it provides.

**PlanAgent:** A PlanAgent has the following capabilities. (1) Maintaining Process Repository (PR) that stores the process plan information. A PlanAgent is responsible for creating new process plans or update existing plans based on user requirements and newly discovered services. A process plan is defined by a set of service classes and the connection relationships among them. (2) Taking a user's service request to generate an execution plan, which is the combination of all chosen process plans. (3) After the execution plan is generated, the PlanAgent is in charge of generating a specification of composite service by conversation with the InfoAgent, a CompositionAgent and its CS-context. Then, the specification is submitted to the CompositionAgent.

**CompositionAgent:** The CompositionAgent's role is to trigger the specification of the composite services and monitor the deployment of this specification. When a CompositionAgent is created, it identifies the first Web services to be triggered according to CS-context. For the sake of simplicity, we assume that the Web services constitute a sequence. When the first Web service is identified, it interacts with the ManagementAgent of this Web service in the objective to ask for service instantiation. If the ManagementAgent agrees on the instantiation after checking the WS-context, ServiceAgent and IS-context of this Web service are both created.

While the service instance is being performed, it identifies the next Web service that is due for execution after current service instance, and engages in conversations with next service's ManagementAgent in the objective to trigger the next Web service.

**ManagementAgent:** A Web service as a component is instantiated when it participates in a new composition. Prior to any instantiation, several elements related to the Web service are checked. These elements constitute a part of the context, denoted by WS-context, of the Web service and are as follows: (1) number of service instances currently running versus maximum number of service instances that can be simultaneously run, (2) execution status of each service instance deployed, and (3) request time of the service instance versus availability time of the service instance.

The role of the ManagementAgent is to track the multiple Web services of type in-instance. It processes the requests of instantiation that are submitted to a Web service and decides if these Web service will join the composition process after checking its WS-contexts. In case of a positive decision, Web service instances, ServiceAgents, and IS-contexts are all deployed by it. An authorization of joining a composite service can be rejected because of multiple reasons: period of nonavailability, overloaded status, or exception situation.

**ServiceAgent:** The ServiceAgent initiates the execution of the service instance and notifies the ManagementAgent about the execution status. Each service instance has its own ServiceAgent and IS-context. While ServiceAgents are in charge of executing the service instances, at the same time, it requests from CompositionAgent to engage in conversations with ManagementAgents of the next Web services to ensure that these next Web services are getting ready for execution. Because of the regular notifications between ServiceAgents and ManagementAgents, exceptional situations are immediately handled so that corrective actions are carried-out on time.

### 3.2 The Interaction Model

A conversation between agents is a sequence of messages that involve two or more participants who intend to achieve a particular purpose. Figure 2 presents the proposed interaction procession.

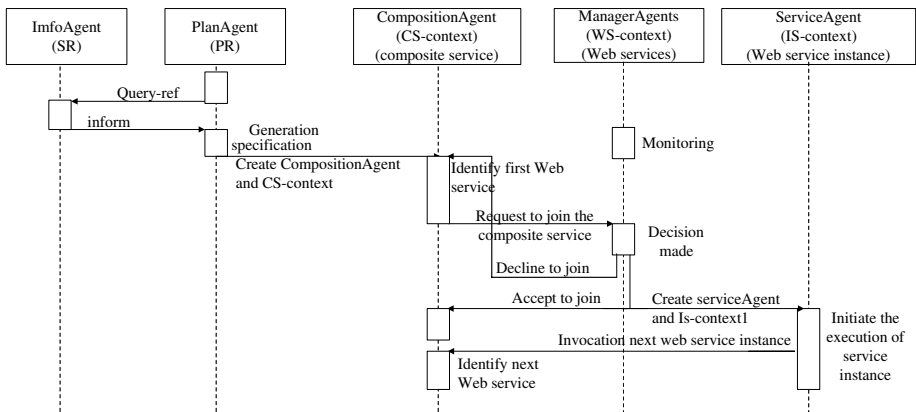


Fig. 2. An Interaction Model of service composition participants

- (1) User submits the request to the PlanAgent. The PlanAgent generate an execution plan based on user’s service request. Afterwards, the PlanAgent creates a CompositionAgent and its CS-context, and the specification of composite service is transmitted to the CompositionAgent.
- (2) The CompositionAgent starts the composition process.It identifies the first Web service and engages in conversations with the ManagementAgent.
- (3) Initially, the ManagementAgent is

in a monitoring mode in which it tracks the instances of this Web service that are currently participating in different composite services. When it receives a request to create an additional instance, it enters the assessment state to evaluate the request. If the request is declined, it sends rejection message to the CompositionAgent, then CompositionAgent select another Web service, which is a substitute of this Web service, to join the composite service. If the request is accepts, it informs its acceptance to the CompositionAgent and creates this Web service's ServiceAgent and the IS-context. (4) The serviceAgent initiates the execution of the service instance and notifies the ManagementAgent about the execution status. Then, the ServiceAgent and the CompositionAgent take each a conversation state. In which, activities to request the participation of the next Web services are performed.

We assume that the Web service, which takes an execution state currently, is denoted as service<sub>*i*</sub> and the next Web service is denoted as Web service<sub>*j*</sub>. When the execution of Web service instance<sub>*i*</sub> is completed, ServiceAgent<sub>*i*</sub> informs the CompositionAgent about that. The CompositionAgent interacts with ServiceAgent<sub>*j*</sub> so that the newly-created instance of Web service<sub>*j*</sub> is triggered and enters the execution state. At the same time, the CompositionAgent initiates conversations with the ManagementAgents of the next Web services that follow Web service<sub>*j*</sub>. In this way, the service composite process is implemented until last Web service is done.

## 4 An Example for Data Format Adaptation

To validate the design of our model, we built a service for real-time multimedia stream playback to a wireless client. Here we discuss our experience and show how the agent system facilitates building composite services that make the data format of the source multimedia stream to adapt a wireless client. The example application is delivering an MPEG video/audio stream to a mobile user using a laptop with both WaveLAN and Ethernet interfaces. The client is only capable of playing RealVideo and RealAudio format.

In this scenario, because the end client cannot play any MPEG stream, we must provide a composite service to transcode MPEG to RealVideo on the fly. Four services were assembled at a composite service to meet the need of data format adaptation. The first service, MPEG Demultiplexer (mpegdemux), is a simple program which separates an integrated MPEG Audio/Video stream into separate MPEG video and MPEG layer 3 (mp3) audio streams. The second service, MPEG video decoder (mpegdec) is used to decode MPEG video stream into raw YUV video frames. The third service is a simple wrapper around the popular command line mp3 decoder called mpg123. It takes in mp3 audio and outputs sound in PCM format. The last service needed for format conversion is a RealVideo encoder. It is developed using a toolkit called RealProducer. We wrapped the video/audio codec in a service, which takes in an audio stream of PCM samples and a video stream of YUV frames, and produces a combined A/V stream in RealVideo format.

Once the input and output types of the services are encoded properly, the System automatically finds a transcoding data path. Depending on the bandwidth available, a compression service may be inserted between MPEGDecoder and RealEncoder to reduce the data rate. Similarly, an FEC encoder and decoder pair may be inserted if the composite service between the RealEncoder and the client includes a wireless link.

## 5 Conclusions

The suggested agent-based dynamic services composition framework can combine a service-oriented approach and multi-agent systems to provide continuous and smooth provision of media services for wireless clients. Five types of agents have been put forward and these agents work with each other to meet user's tasks. During the composition process, the different agents are aware of the context of their respective services in the objective to devise composite services on-the-fly.

For developing a proof-of-concept implementation of our proposal, we are investigating the use of Jini (<http://www.jini.org/>), which is a Java-oriented service-oriented system. Our future work therefore consists of completing the implementation of our system architecture and demonstrating it on real-life examples.

## References

1. M.Sheshagiri, M.desJardins, and T.Finin.: A planner for composing services described in DAML-S. In Proceedings of the AAMAS Workshop on Web Services and Agent-based Engineering, 2003
2. D.Wu, B.Parsia, E.Sirin, J.Hendler, and D.Nau.: Automating DAML-S Web Services composition using SHOP2. In Proceedings of the 2nd International Semantic Web Conference, ISWC2003, Sanibel Island, Florida, USA, October 20-23, 2003, 2003
3. Le-Hung Vu, Manfred Hauswirth, Karl Aberer.: Towards P2P-based Semantic Web Service Discovery with QoS Support. Proceeding of Workshop on Business Processes and Services (BPS), Nancy, France, 2005
4. S. Thakkar, C. A. Knoblock, J.L.Ambite, and C.Shahabo.: Dynamically composing Web services from online sources. In Proceeding of 2002 AAAI Workshop on Intelligent Service Integration, Edmonton, Alberta, Canada, 2002
5. Zakaria Maamar, Soraya Kouadri Moste faoui, and Hamdi Yahyaoui.: Toward an Agent-Based and Context-Oriented Approach for Web Services Composition. In IEEE Transactions on Knowledge and Data Engineering, VOL. 17, NO. 5, MAY 2005

# A Multi-subset Possible World Semantics for Intention Operator of Agent

Shan-Li Hu<sup>1,3</sup> and Chun-Yi Shi<sup>2</sup>

<sup>1</sup>Dept. of Computer Science and Technology, Fuzhou University, Fuzhou 350002, China  
husl@fzu.edu.cn

<sup>2</sup>Dept. of Computer Science and Technology, Tsinghua University, Beijing 100084, China  
scy@tsinghua.edu.cn

<sup>3</sup>Laboratory of Computer Science, Chinese Academy of Sciences, Beijing 100080, China

**Abstract.** Intentions, a crucial part of the mental state of an agent, play an important role in determining the behavior of rational agents. In order to eliminate the flaws with existing logic of intention, in this paper, we address the requests for intention semantics on formal frameworks of rational agents and the problems with existing logic of intention, put forward a novel semantics for intention, called the multi-subset possible world semantics, and its application in the formalization of intention for agent. The multi-subset possible world semantics use three subsets to describe intention. It not only avoids the logical omniscience problem and other related problems (such as side-effect problem, and etc) but also overcomes the shortcomings of the true-false subset semantics and twin-subset semantics. Compared with Konolige and Pollack's model of intention, this semantics model doesn't lose the reasoning ability of non-equivalent intention, and what's more, it is simpler, more natural and satisfies the K axiom and the Joint Consistency. Our framework invalidates the problematic properties with existing logic of intention. And by imposing certain constraints on the algebraic structure of the models, many desirable properties can be obtained. At last we make an analysis for the multi-subset possible world semantics. Actually the multi-subset possible world semantics provides a new method for semantic representation of non-normal modal operators. It can be used in establishing new proper agent's logic systems.

**Keywords:** Agent, intention, semantics, multi-subset possible world semantics.

## 1 Introduction

In a multi-agent system, how to abstract and model an agent has become an important subject in the field of AI (Artificial Intelligence), Software Engineering, etc. In AI domains, people often study agents from mental stance, generally consider belief, desire and intention as the basis attributes of mentality (Abbreviated as BDI). The logic of belief, desire and intention has been paid attention to in the documents of AI on the design of rational agent [1], [2], [3], [4], [5]. Its importance lies in its action as the logical framework of formal agent. This logic needs a formal foundation on which is created the epistemic logic that refers to knowledge and belief. Therefore a large amount of researches are usually based on the method of normal possible world. People generally model belief desire and intention to normal modal operators [2], [3],



[4], [5]. Thus, unavoidably, some intrinsic questions of the method will exist. However, intention, an indispensable conscious state attribute in abstracting and modeling an agent, plays an important part in determining the behaviors of the rational agent. Intention determines the choice of the future action, thus keeps the self-balance of belief, goal, plan and behavior of a limited autonomy system. Recently, study in the formalization of the intention has already become an important, common concerned subject. There are important applications in practical reasoning, cognitive modeling, natural language processing and the plan, negotiation, cooperation, competition etc. among agents' systems. The following are some major problems existing in modeling the intention to a normal modal operator. (Here,  $I$  is intention operator)

- (1) Logical omniscience problem:  $\models \varphi \Rightarrow \models I(\varphi)$
- (2) The side-effect problem of tautological implication:  $\models \varphi \rightarrow \psi \Rightarrow \models I(\varphi) \rightarrow I(\psi)$
- (3) Disjunction magnification problem:  $\models I(\varphi) \rightarrow I(\varphi \vee \psi)$
- (4) Conjunction separation problem:  $\models I(\varphi \wedge \psi) \rightarrow I(\varphi) \wedge I(\psi)$

It is known that these problems are not acceptable during the practical applications. Classic normal modal logic isn't suitable to act as the formal tool of agents' intention [3], [4], [5], [6].

Directed against the above problems, Konolige and Pollack[3] proposed an intentional description theory(called KP model), which used non-normal modal logic as the description tool, and its semantics modal is a triple  $M = \langle W, \Sigma, I \rangle$ , where  $W$  is a set of possible worlds. To every world,  $w \in W$ , there is an evaluation function, which determines the value of sentences in language  $L$ .

To a formula  $\varphi$ , the set of possible worlds, which make  $\varphi$  true is marked as  $M_\varphi$ ,  $M_\varphi =_{df} \{w \in W \mid W, w \models \varphi\}$ .  $\Sigma \subseteq W$  is used to explain belief, the semantics of the belief modal operator  $B$  is

$$\langle W, \Sigma, I \rangle \models B(\varphi) \text{ iff } \forall w \in \Sigma \text{ such that } W, w \models \varphi, \text{ that is } \Sigma \subseteq M_\varphi$$

$I \subseteq P(W)$  ( $P(W)$  is the powerset of  $W$ ) is used to explain an intention, the semantics of an intention modal operator  $I$  is

$$\langle W, \Sigma, I \rangle \models I(\varphi) \text{ iff } \exists I_\varphi \in I \text{ such that } I_\varphi = M_\varphi, \text{ that is } M_\varphi \in I.$$

Actually, every element  $I_\varphi$  in  $I$ , that is  $M_\varphi$ . The only different from normal modal semantics explanation is that they use “ $=$ ” instead of “ $\subseteq$ ” in the above explanation regulation, thus the side effect of logical implication was avoided.

But, in the KP model, there are some problems as follows:

- 1) In KP model, if  $I(\varphi)$  can be deduced by  $I(\psi)$  then  $\varphi \equiv \psi$ , hence the intention inference disappears in this model. That is, this model lost the reasoning ability for the non-equivalent intentions. Thereby, it is inadvisable.
- 2) KP model doesn't satisfy K axiom and Joint Consistency principle (Section 3). So, many scholars used classical but non-normal modal logic in the formalization of intention for agents [6]. But non-normal possible worlds, the foundation of classical but non-normal modal logic, are inconceivable, such as proposition  $\varphi$  and its negative proposition  $\neg\varphi$  can be true at the same time in non-normal

possible worlds. And the true value of a compound formula can't be evaluated by the value of each atom propositions. It can be either true or false. It is unacceptable to use this incredible semantics explanation to formalize the realistic system closely related to people's life.

Therefore, it is necessary to find out a formal semantics that is believable directly perceived through the sense and can describe intention properly.

## 2 A Multi-subset Possible World Semantics for Intention and Its Intuitionist Explanation

In order to emphasize the description of intention semantics and its difference with KP system, we do not involve desire and goal in this paper. Simply, the formalization system here is the expansion of classical logic. It extended the possible world framework by introducing the corresponding modal operators of belief and intention. But the intention operator is non-normal, and it uses multi-subset possible world semantics we developed. So this system is a simple hybrid modal logical system.

**Definition 1.** (Multi-modal Language L) Supposing,  $Prop = \{p, q, \dots\}$  is countable atomic proposition sets. Well-formed formula sets  $FML$  definition by recursion is as follows:

- (1) if  $p \in Prop$ , then  $p \in FML$ ,
- (2) if  $\phi, \psi \in FML$ , then  $\neg\phi \in FML$  and  $\phi \vee \psi \in FML$ ,
- (3) if  $\phi \in FML$ , then  $B(\phi) \in FML$  and  $I(\phi) \in FML$ .

The semantics of operators '¬'(not) and '∨'(or) are the same as classical proposition logic. And other definitions of connectives in classical proposition logic are omitted here. In language L, the modal operator  $B$  and  $I$  are called belief operator and intention operator separately. Formula  $B(\phi)$  is called "Agent believes  $\phi$ ", while formula  $I(\phi)$  is called "Agent intends  $\phi$ ".

**Definition 2.** (Semantic Model  $M$ ) Agent's semantic model  $M$  is a tuple,  $M = \langle W, R_B, R_{IT}, R_{IF1}, R_{IF2}, V \rangle$ , where  $W$  is a non-empty set, and can be considered as sets of possible world.  $R_B, R_{IT}, R_{IF1}, R_{IF2} \subseteq W \times W$ . To every  $w \in W, R_B, R_{IT}, R_{IF1}, R_{IF2}$  mapping  $w$  to the attainable world sets of  $w$ 's belief, intention(true), intention(false1), intention(false2), marked  $R_B(w), R_{IT}(w), R_{IF1}(w), R_{IF2}(w)$ ; They are non-empty subsets of  $W$ , and  $R_{IT}(w), R_{IF1}(w), R_{IF2}(w)$  are disjoint among each other. And  $V$  is evaluation function.

$$V : W \times Prop \rightarrow \{1, 0\}$$

To formula  $\phi, \phi$  is true or valid in  $w$ , iff  $V(w, \phi) = 1$ , mark it as  $\langle M, w \rangle \models \phi$ . We omit  $v$  in the discussion because it is not important. The explanations of increased modal operator  $B$  and  $I$  are given in the following definition.

- $\langle M, w \rangle \models p$  where  $p \in Prop$ , iff  $V(w, p) = 1$
- $\langle M, w \rangle \models \neg\phi$ , iff  $\langle M, w \rangle \not\models \phi$
- $\langle M, w \rangle \models \phi \vee \psi$ , iff  $\langle M, w \rangle \models \phi$  or  $\langle M, w \rangle \models \psi$

$\langle M, w \rangle \models B(\varphi)$ , iff  $\forall w' \in W$  : if  $w' \in R_B(w)$  then  $\langle M, w' \rangle \models \varphi$

$\langle M, w \rangle \models I(\varphi)$ , iff

- (1)  $\forall w' \in W$ , if  $w' \in R_{IT}(w)$  then  $\langle M, w' \rangle \models \varphi$ ; and
- (2)  $\exists w_1'' \in W$ , such that  $w_1'' \in R_{IFI}(w)$  and  $\langle M, w_1'' \rangle \models \neg\varphi$ ; and
- (3)  $\exists w_2'' \in W$ , such that  $w_2'' \in R_{IF2}(w)$  and  $\langle M, w_2'' \rangle \models \neg\varphi$ .

The intuitionistic explanation of belief and intention semantics: in world  $w$ , belief represents agent's preference to some possible worlds (that is  $R_B(w)$ ). Agents merely believe the propositions, which are true in all these possible worlds. Moreover, intention represents agent's attention to some possible worlds. But what different from belief is that these worlds are separated into three disjoint parts among each other. These three parts are marked as  $R_{IT}(w)$ ,  $R_{IFI}(w)$ , and  $R_{IF2}(w)$ . Agent considers the propositions that are true in  $R_{IT}(w)$  to be possibly realize, while considers the ones, which has not realized and not consequentially realize, to be partly false in  $R_{IFI}(w)$ , and  $R_{IF2}(w)$ . Obviously, only the propositions, which have not been realized, not been consequentially realized, and can be possibly realized, are worthy to be intended by rational agents.

In world  $w$ , belief in the definition only uses one subset of  $W$  (that is  $R_B(w)$ ) to describe, and intention use three disjoint subsets among each other (that is  $R_{IT}(w)$ ,  $R_{IFI}(w)$ , and  $R_{IF2}(w)$ ) to describe rather than using the subset of  $W$ 's powerset. Thereby, compared with KP model, it is simpler, more natural, and does not lose the reasoning ability for non-equivalent intention. Of course, the expected properties will be proved in the following.

### 3 The Rationality of This Model

Bratman[1] put forward that restricted autonomy system must satisfy the necessary of rationality: "asymmetric proposition" and "no side-effect principle". The asymmetric proposition is constituted by consistency and imperfection. The form is expressed:

$$\text{Consistency} \quad \forall M. M \models I(\varphi) \wedge B(\neg\varphi) \quad (\text{AI1})$$

$$\text{Imperfection} \quad \exists M. M \models I(\varphi) \wedge \neg B(\varphi) \quad (\text{AI2})$$

$$\text{No side-effect principle} \quad \exists M. M \models (\varphi \rightarrow \psi) \wedge I(\varphi) \wedge \neg I(\psi) \quad (\text{AI3})$$

Besides to "asymmetric proposition" and "no side-effect principle", belief and intention should satisfy the following property:

$$\text{Non-transfer principle [7]} \quad \exists M. M \models B(\varphi) \wedge \neg I(\varphi) \quad (\text{AI4})$$

$$K \text{ axiom:} \quad \forall M. M \models I(\varphi \rightarrow \psi) \rightarrow (I(\varphi) \rightarrow I(\psi)) \quad (\text{AI5})$$

$$\text{Consistency principle:} \quad \forall M. M \models \neg I(\varphi \wedge \neg\varphi). \quad (\text{AI6})$$

$$\text{Joint consistency principle:} \quad \forall M. M \models I(\varphi) \wedge I(\psi) \rightarrow I(\varphi \wedge \psi) \quad (\text{AI7})$$

$$\text{Joint imperfection principle:} \quad \exists M. M \models I(\varphi \vee \psi) \wedge \neg I(\varphi \wedge \psi) \quad (\text{AI8})$$

$$\text{and } \exists M. M \models I(\varphi \wedge \psi) \wedge \neg I(\varphi) \wedge \neg I(\psi) \quad (\text{AI9})$$

$$\text{nontrivial principle:} \quad \forall M. M \models I(\varphi) \rightarrow \neg B(\varphi) \quad (\text{AI10})$$

Because  $K$  axiom exists in normal modal logical system, non-normal modal logical system and modal logical system of epistemology, it is comprehensible that intention satisfies  $K$  axiom: Suppose rational agent intends  $\phi$ , and  $\phi \rightarrow \psi$ , then of course it should intends  $\psi$ , otherwise it didn't need to intent  $\phi \rightarrow \psi$ .

Obviously, intention must satisfy consistency principle AI6, because rational agent should not have opposite intentions. Intention should satisfy Joint Consistency principle AI7 because if agent intends  $\phi$  and  $\psi$ , then of course it intends  $\phi \wedge \psi$ . Note that it is different from AI8, which means that agent intends  $\phi$  or  $\psi$ , but does not always intent  $\phi \wedge \psi$ ; and it is also different from AI9, which means that agent intends  $\phi \wedge \psi$ , but does not necessarily intent  $\phi$  and intent  $\psi$ . We should pay attention to their differences. And we call AI8 and AI9 the Joint Imperfection principle. Belief and intention should also satisfy nontrivial principle AI10. Because if rational agent intends  $\phi$ , then it doesn't believe  $\phi$  can be realized automatically, otherwise it needn't intent  $\phi$ .

Since we use three subsets  $R_{I1}(w), R_{I1'}(w), R_{I2}(w)$  to describe intention, so to intention, there is no logical omniscience problem mentioned in the preface. Besides, it is easy to prove that: properties AI3, AI5-AI9 are true in model  $M$ . and the problems such as side effects of logical implication mentioned in the preface, etc. don't exist for intention. That is we have:

**Theorem 1.** In model  $M$ , The four problems of logic omniscience etc mentioned before don't exist. And properties AI1, AI2, AI10 are true here.

In order to express the relationship between belief and intention, and make intention satisfy the properties related to belief, we should impose certain constrains on the algebraic structure of the model. Thus, we propose some constraint conditions called achievability and nontrivial.

**Definition 3.** Model  $M = \langle W, R_B, R_{I1}, R_{I1'}, R_{I2}, V \rangle$  is admissible iff it is achievable:  $\forall w \in W. R_B(w) \cap R_{I1}(w) \neq \emptyset$  and it is nontrivial:  $R_{I1}(w) \subseteq R_B(w)$  and  $R_{I2}(w) \subseteq R_B(w)$ .

**Theorem 2.** In admissible model, properties AI1, AI2, AI10 is true.

### 4 Comparison and Conclusion

In this paper, we discussed the semantic requirement of intention and proposed a new intention model. This model has intuitionistic semantic explanation, which is based on normal possible world, and can satisfy the requirement for intention semantic (AI1-AI10). Compared with the work of Cohen and Levesque [2], it avoided "the logical omniscience" problem and other related problems (such as side-effect problem). Compared with the work of Konolige and Pollack [3], the description of intention is simple, natural, and its semantic explanation is based on the normal possible world, and satisfies  $K$  axiom and Joint Consistency principle (AI7), and doesn't lose the reasoning ability for non-equivalent intention.

The KP model proposed by Konolige and Pollack [3], does not satisfy  $K$  axiom AI5. Because if  $M_\phi \in I$  and  $M_{\phi \rightarrow \psi} \in I$ , then there is not always  $M_\psi \in I$ . And KP model doesn't satisfy Joint Consistency principle (AI7) neither. Because, if  $M_\phi \in I$  and  $M_\psi \in I$ , then there is not always  $M_{\phi \wedge \psi} \in I$ .

The multi-subset possible world semantic proposed in this paper not only overcomes the shortcomings that agent can't intent both  $\varphi \rightarrow \psi$  and  $\varphi$  at the same time in the true-false semantic [8], but also overcomes the shortcomings that it can't satisfy the Joint Consistency principle (A19:  $\exists M$  such that  $M \models I(\varphi \wedge \psi) \wedge \neg I(\varphi) \wedge \neg I(\psi)$ ) in twin-subset semantic [9]. It is more natural than improved twin-subset semantic [10]. The specific discussion is omitted here. Besides, this method is universal and can be applied in not only the formal description of belief, desire in agent's BDI structure, but also the generic non-normal modal operators.

The above discussion shows that the multi-subset possible world semantics provides a new suitable description of semantics for non-normal modal operator. It is an important development of the possible world semantics of classical normal modal operators and is a useful tool for the formalization of rational agent's properties. It can be applied to create a new appropriate logical system for agents.

**Acknowledgments.** This paper is supported by the National Natural Science Foundation of China under Grant Nos. 60373079, 60573076.

## References

1. Bratman M E. Intentions, Plans and Practical Reason. Cambridge, MA: Harvard University Press, 1987
2. Cohen P R, Levesque H J. Intention is Choice with Commitment. *Artificial Intelligence*, 1990,42(2-3): 213-261
3. Konolige K, Pollack M E. A Representationalist Theory of Intention. In: Bajcsy R ed. Proceedings of the Thirteenth International Joint Conference on Artificial Intelligence. San Mateo, CA: Morgan Kaufmann Publishers, Inc., 1993.390-395
4. Rao A S, Georgeff M P. Modeling rational Agents within a BDI architecture. In: Allen J, Fikes R, Sandewall E eds. Principles of Knowledge Representation and Reasoning: Proceedings of the Second International Conference (KR-91). San Mateo, CA: Morgan Kaufmann Publishers, Inc., 1991.473-484
5. Rao A S, Georgeff M P. The Semantics of Intention Maintenance for Rational Agents. In: Mellish S C ed. Proceedings of the 14<sup>th</sup> International Joint Conference on Artificial Intelligence. San Mateo, CA: Morgan Kaufmann Publishers, Inc, 1995. 704-710
6. Cavedon L, Padgham L, Rao A et al. Revisiting rationality for agents with intentions. In: Xin Yao ed. Proceedings of the 8<sup>th</sup> Australian Joint Conference on Artificial Intelligence. Singapore: World Scientific Publishing Co. Pet. Ltd. 1995.131-138
7. Rao A S, Georgeff M P. Asymmetry thesis and side-effect Problems in linear- time and branching-time intention logic. In: Proceedings of the 12th International Joint Conference on Artificial Intelligence. San Mateo, CA: Morgan Kaufmann Publishers, Inc., 1991.498-504
8. HU Shan-Li, SHI Chun-Yi. Agent Logic and The True-false Subset Semantics. *Journal of Software*, 2002,13(11): 2112-2115 (in Chinese)
9. HU Shan-Li, SHI Chun-Yi. Twin-Subset Semantic Model for Intention. On the Proceedings of the Second International Conference on Machine Learning and Cybernetics, IEEE volume 4 (2003.11):2004-2008
10. HU Shan-Li, SHI Chun-Yi. An Improved Twin-Subset Semantic Model for Intention of Agent. *Journal of Software*, 2006,17(3): 396-402 (in Chinese)

# A Concurrent Agent Model Based on Twin-Subset Semantic

Youmin Ke<sup>1,2</sup> and Shanli Hu<sup>1,2</sup>

<sup>1</sup>Department of Computer Science and Technology,  
Fuzhou University, Fuzhou 350002, China

<sup>2</sup>Laboratory of Computer Science, Chinese Academy of Sciences,  
Beijing, 100080, China  
henrykym@126.com, husl@fzu.edu.cn

**Abstract.** Under the multi-agent system (MAS) circumstances, the concurrent behaviors turn out to be very important. Concurrent behaviors can be divided into irrelative and correlative, most of them are correlative and agents should be able to deduce their behaviors. This paper put forward a concurrent Agent model based on twin-subset semantic. This model aims at the characteristics of concurrent behaviors in MAS. Based on twin-subset semantic, the logical omniscience problem and other related problems (such as side-effect problem) can be avoided. This model divides the time set into macro-time and micro-time sets, and describes the concurrency in macro-time level. In this model, parallel actions in macro-time are interleaving in micro-time level. It can be applied to establish the logical foundation for the cooperation and competition based on concurrency in MAS.

## 1 Introduction

Agent modeling is one of the main aspects of multi-agent system (MAS) researches. Cohen & Levesque's linear-time model [1] and Rao & Georgeff's branching-time model [2] have far-reaching effects on the future researches. Belief, desire and intention have been widely concerned in most AI literatures on designing rational agents. They are used as the foundations of logical framework. Because of this, many researchers tend to use normal possible worlds on their methods, and the consequent problems are unavoidable. As an indispensable consciousness attribute in agent's modeling, intention plays an important part in determining the behaviors of the rational agent. Modeling intention as a normal modal operator will give birth to the following problems. Here intention is denoted by *INT*.

- (1) logical omniscience problem:  $\models \varphi \Rightarrow \models INT(\varphi)$
- (2) the side-effect problem:  $\models \varphi \rightarrow \psi \Rightarrow \models INT(\varphi) \rightarrow INT(\psi)$
- (3) disjunction magnification problem:  $\models INT(\varphi) \rightarrow INT(\varphi \vee \psi)$
- (4) conjunction separation problem:  $\models INT(\varphi \wedge \psi) \rightarrow INT(\varphi) \wedge INT(\psi)$

People have realized these problems are unacceptable in practical use; therefore classic normal modal logic is not suitable to be a formalization tool of agent's

intention. We need to search for a formal semantic that is intuitionistic, trustable and can correctly represent agent's intention. Hu's improved twin-subset semantic model [9] formalizes intention into modal operators in two-value logic. The model is still based on possible world semantics but can avoid the above problems.

Under the MAS circumstances, the concurrent behaviors turn out to be very important. Concurrent behaviors can be divided into irrelative and correlative. Agents that are carrying out the irrelative concurrent behaviors needn't to think about those behaviors which other agents performed at the same time, but under the circumstances of collaborations or competitions in MAS, most are correlative and agents should be able to deduce their behaviors. De Vries W etc [7] gave a truly concurrent semantic agent model, but there were no solutions to the non-deterministic and behaviors conflicts induced by truly concurrency. Compared with the overlapping concurrent model, this model itself was not brief enough. Wang etc [8] gave an overlapping concurrent BDI-Agent model to describe concurrency, but there still exist the above four problems.

This paper put forward a concurrent Agent model based on twin-subset semantic. This model aims at the characteristics of concurrent behaviors in MAS. Based on twin-subset semantic, the logical omniscience problem and other related problems (such as side-effect problem) can be avoided. This model divides the time set into macro-time and micro-time sets, and describes the concurrency in macro-time level. It can be applied to establish the logical foundation for the cooperation and competition based on concurrency in MAS, and also can advance the work of Hu and de Vries and etc.

## 2 Model

### 2.1 Syntax

Like Rao&Georgeff's model [2], each world's temporal structure is called a time tree. Formulas are divided into state formulas and path formulas. State formula evaluates the state of a time point in the time tree. Path formula evaluates the appointed path in the time tree and two defined modal operators *optional* and *inevitable* act on path formulas. The elementary operators include: fundamental propositional symbol  $\neg$ ,  $\vee$  and existential quantifier  $\exists$ . There are modal operators *BEL*, *DES* and *INT* as well as standard temporal operators  $\bigcirc$ (next),  $\diamond$ (eventually),  $\square$ (always),  $\cup$ (until) act on state or path formulas. These operators can be organized in different ways to represent agent's possible choices. In addition, we introduce concurrent operators:  $|$  links two state formulas and only one of them is chosen to be perform;  $\parallel$  links two state formulas and they perform concurrently;  $;$  links two state formulas and they perform orderly.

State formula is defined as follow:

1. Any first-order formulas are state formulas.
2. If  $\varphi_1$  and  $\varphi_2$  are state formulas and  $x$  is an unit or event variable, then  $\neg\varphi$ ,  $\varphi_1 \vee \varphi_2$ ,  $\exists x\varphi_1(x)$  and  $\varphi_1|\varphi_2$ ,  $\varphi_1\parallel\varphi_2$ ,  $\varphi_1;\varphi_2$  are state formulas.
3. If  $e$  is an event, then *succeeds*( $e$ ), *fails*( $e$ ), *does*( $e$ ), *succeeded* ( $e$ ), *failed* ( $e$ ) and *done*( $e$ ) are state formulas. They represent the state of event  $e$  whether is successful, unsuccessful or no matter successful or not in its future and past.

4. If  $\varphi$  is a state formula, then  $BEL(\varphi)$ ,  $DES(\varphi)$  and  $INT(\varphi)$  are state formulas.
5. If  $\Psi$  is a path formula, then  $optional(\Psi)$  and  $inevitable(\Psi)$  are state formulas.

Path formula is defined as follow:

1. Any state formulas are also path formulas.
2. If  $\Psi_1$  and  $\Psi_2$  are path formulas, then  $\neg\Psi_1, \Psi_1 \vee \Psi_2, \Psi_1 \cup \Psi_2, \diamond\Psi_1, \square\Psi_1$  and  $\bigcirc\Psi_1$  are path formulas.

## 2.2 Semantic

**Definition 1.** Agent model is a tuple

$$M = \langle W, T, MT, \prec, act, R_b, R_d, R_i^T, R_i^F, V \rangle$$

And we use  $U_{ag}$  represents the set of agents;  $U_{ac}$  represents the set of atomic actions.

$W$  is the non-empty set of possible worlds.

$T$  is the set of micro-time points. Each micro-time point has linear history and branch futures.

$\prec \subseteq T \times T$  is binary relation on the time point. We define operators  $<, \leq$  as

$$t_1 < t_2 \text{ iff } t_1 \prec t_2 \text{ or } \exists t' \in T, t_1 < t' < t_2, \text{ and } t_1 \leq t_2 \text{ iff } t_1 = t_2 \text{ or } t_1 < t_2$$

$MT$  is the set of macro-time points.  $MT \subseteq T$ , And  $\forall t_1^M, t_2^M \in MT, \exists t' \in T, t' \notin MT$  such that  $t_1^M < t' < t_2^M$ . That is, there should be some micro-time points between two different macro-time points.

As to  $w \in W, w = \langle T_w, \prec_w, MT_w, act_w \rangle$ , where  $T_w, \prec_w, MT_w, act_w$  are the constraints of  $T, \prec, MT, act$  in  $w$ .

$act$ : action function,  $act: U_{Ag} \times U_{Ac} \rightarrow W$ , and we assume that each action should take at least one macro-time segment, or to say, the duration of the action should contain at least one macro-time point, if not, we can split the macro-time into shorter segments, since we check the states of actions at each macro-time points only.

$R_b: U_{Ag} \times W \times MT \rightarrow W$  The belief-accessible worlds at macro-time point

$R_d: U_{Ag} \times W \times MT \rightarrow W$  The desire-accessible worlds at macro-time point

$R_i^T: U_{Ag} \times W \times MT \rightarrow W$  The true-intention-accessible worlds at macro-time point.

$R_i^F: U_{Ag} \times W \times MT \rightarrow W$  The false-intention-accessible worlds at macro-time point.

$V: W \times Prop \rightarrow \{0,1\}$ , to a formula  $\varphi$ ,  $\varphi$  is true or valid in  $w$  iff  $V(w, \varphi) = 1$ , denoted by  $\langle M, w \rangle \models \varphi$ . Since  $V$  is not important in our discussion, it can be omitted in the following discussion.

**Definition 2.** For any  $w \in W$

$$\langle M, w \rangle \models p \text{ where } p \in Prop, \text{ iff } V(w, p) = 1$$

$$\langle M, w \rangle \models \neg\varphi \text{ iff } \langle M, w \rangle \not\models \varphi$$

$$\langle M, w \rangle \models \varphi \vee \psi \text{ iff } \langle M, w \rangle \models \varphi \text{ or } \langle M, w \rangle \models \psi$$

$$\langle M, w \rangle \models BEL(\varphi) \text{ iff } \forall w' \in W \text{ if } w' \in R_b(w) \text{ then } \langle M, w' \rangle \models \varphi$$



$\langle M, w \rangle \models DES(\varphi)$  iff  $\forall w' \in W$  if  $w' \in R_d(w)$  then  $\langle M, w' \rangle \models \varphi$   
 $\langle M, w \rangle \models INT(\varphi)$  iff

- (1)  $\forall w' \in W$  if  $w' \in R_i^T(w)$  then  $\langle M, w' \rangle \models \varphi$ ; and
- (2)  $\exists w_1'' \in W$  such that  $w_1'' \in R_i^F(w)$  and  $\langle M, w_1'' \rangle \models \neg\varphi$ ; and
- (3)  $\exists w_2'' \in W$  such that  $w_2'' \neq w_1''$ ,  $w_2'' \in R_i^F(w)$  and  $\langle M, w_2'' \rangle \models \neg\varphi$

The explanation of semantics of belief and intention: in the world  $w$ , agent believes the propositions if they are true in some possible worlds (that is  $R_b(w)$ ). Like belief, intention also shows agent's preference to some possible worlds but these possible worlds are divided into two disjoint parts  $R_i^T(w)$  and  $R_i^F(w)$ . In  $R_i^T(w)$ , the propositions are true if agent thinks they are able to be achieved, and false in some worlds of  $R_i^F(w)$  (at least two:  $w_1''$ ,  $w_2''$ ) if agent thinks they have not yet been achieved and will not be necessarily achieved. It is obvious that only the propositions that have not yet been achieved and will not be necessarily achieved are worth intending by rational agents [9].

**Definition 3.**  $\forall c, d \in U_{Ac}$ ,  $c \parallel d$  represents  $c$  and  $d$  perform concurrently at a certain micro-time segment.  $c \mid d$  represents  $c$  or  $d$  is chosen to perform at a certain micro-time segment. And  $c; d$  represents  $d$  performs after  $c$  finished.

For model  $M$ , possible-worlds  $W$ , macro-time point  $t$ , proposition  $p, q$  and individual agents  $x, y$ , the semantics of the state formulas are:

$\langle M, w, t \rangle \models INT(x, p) \parallel INT(x, q)$  iff  $\exists t_1^M, t_2^M \in MT, t_1^M \prec t_2^M$  and  $\nexists t_3^M \in MT$ , such that  $t_1^M < t_3^M < t_2^M$ ,  $\exists t_1', t_2' \in T$ , and  $t_1^M \leq t_1', t_2' \leq t_2^M, t_1' \neq t_2'$  such that  $act(x, p, t_1')$  and  $act(x, q, t_2')$ , that is agent will take actions on the micro-time points between two adjacent macro-time points  $t_1^M, t_2^M$ .

$\langle M, w, t \rangle \models INT(x, p); INT(x, q)$  iff  $\exists t_1^M, t_2^M \in MT, t_1^M \prec t_2^M$  and  $\nexists t_3^M \in MT$ , such that  $t_1^M < t_3^M < t_2^M$ ,  $\exists t_1', t_2' \in T$ , and  $t_1^M \leq t_1' < t_2' \leq t_2^M$ , such that  $act(x, p, t_1')$  and  $act(x, q, t_2')$ , that is agent will take actions successively on micro-time points between two adjacent macro-time points  $t_1^M, t_2^M$ .

$\langle M, w, t \rangle \models INT(x, p) \mid INT(x, q)$  iff  $\exists t_1^M, t_2^M \in MT, t_1^M \prec t_2^M$  and  $\nexists t_3^M \in MT$ , such that  $t_1^M < t_3^M < t_2^M$ ,  $act(x, p, t_1')$  or  $act(x, q, t_2')$  that is agent will take actions on one of micro-time points between two adjacent macro-time points  $t_1^M$  and  $t_2^M$ .

$\langle M, w, t \rangle \models done(x, a)$  iff  $\exists t' \in MT, t' < t, Act(t', t) = act(x, a)$ , that is action  $a$  occurred on the previous macro-time point  $t'$  and finished at the current time point  $t$ .

$\langle M, w, t \rangle \models does(x, a)$  iff  $\exists t' \in MT, t < t', Act(t, t') = act(x, a)$ , that is action  $a$  occurs between the current time point  $t$  and the next macro-time point  $t'$ .

Like the *INT* operators, the semantics of concurrent operators of the operators *done* and *does* operators can be defined by analogy.

In our assumption, an action's duration contains at least one macro-time point, therefore, the start and end of an action will not be between the same adjacent macro-time points, that is there will not be actions out of being detected.

### 3 Basic Properties of Model

Bratman [4] proposed the necessary condition of rationality, which a finite autonomous system must satisfy: "asymmetry proposition" and "non-side-effect principle". The forms of related principles are expressed as follow:

$$\text{Consistency} \quad \forall M, M \not\models INT(\varphi) \wedge BEL(\neg\varphi) \quad (AI1)$$

$$\text{Imperfection} \quad \exists M, M \models INT(\varphi) \wedge \neg BEL(\varphi) \quad (AI2)$$

$$\text{Non-side-effect principle} \quad \exists M, M \models (\varphi \rightarrow \psi) \wedge INT(\varphi) \wedge \neg INT(\psi) \quad (AI3)$$

$$\text{Non-transfer principle} \quad \exists M, M \models BEL(\varphi) \wedge \neg INT(\varphi) \quad (AI4)$$

$$K \text{ axiom:} \quad \forall M, M \models INT(\varphi \rightarrow \psi) \rightarrow (INT(\varphi) \rightarrow INT(\psi)) \quad (AI5)$$

$$\text{Consistency principle:} \quad \forall M, M \models \neg INT(\varphi \wedge \neg\varphi) \quad (AI6)$$

$$\text{Joint consistency principle:} \quad \forall M, M \models INT(\varphi) \wedge INT(\psi) \rightarrow INT(\varphi \wedge \psi) \quad (AI7)$$

$$\text{Joint imperfection principle:} \quad \exists M, M \models INT(\varphi \vee \psi) \wedge \neg INT(\varphi \wedge \psi) \quad (AI8)$$

$$\text{and} \quad \exists M, M \models INT(\varphi \wedge \psi) \wedge \neg INT(\varphi) \wedge \neg INT(\psi) \quad (AI9)$$

$$\text{Nontrivial principle:} \quad \forall M, M \models INT(\varphi) \rightarrow \neg BEL(\varphi) \quad (AI10)$$

As we use two subsets  $R_i^T(w)$ ,  $R_i^F(w)$  to describe intention, so those propositions, which are always true, will not be intended. Hence, there will be no logical omniscience problem mentioned in the preface, and what's more, it is easy to prove that properties AI3, AI5-AI9 hold true in model  $M$  and the other three problems mentioned in the preface can be avoided. And the following theorems hold true. [9]

**Theorem 1.** In model  $M$ , The four problems of logic omniscience mentioned before will not exist and properties *AI1*, *AI2*, *AI10* hold true.

**Definition 4.** Model  $M$  is called adoptable iff its achievable:  $\forall w \in W. R_b(w) \cap R_i^T(w) \neq \emptyset$  and nontrivial:  $R_i^F(w) \subseteq R_b(w)$ .

**Theorem 2.** In such adoptable model  $M$ , properties *AI1*, *AI2*, *AI10* hold true.

### 4 Conclusions

This paper put forward a concurrent Agent model based on twin-subset semantic. This model aims at the characteristics of concurrent behaviors in MAS. Based on twin-subset semantic, the logical omniscience problem and other related problems (such as side-effect problem) can be avoided. This model divides the time set into macro-time and micro-time sets, and describes the concurrency in macro-time level. It can be

applied to establish the logical foundation for the cooperation and competition based on concurrency in MAS, and also can advance the work of Hu and de Vries and etc.

**Acknowledgements.** This paper is supported by the National Natural Science Foundation of China under Grant No.60373079, No.60573076.

## References

1. Cohen P, Levesque H. Intention is choice with commitment. *Artificial Intelligence*, 1990,42(2-3):213~261.
2. Rao AS, Georgeff MP. Modeling rational agents within a BDI-architecture. In: Allen J, Fikes R, Sandewall E, eds. *Proceedings of the 2rd International Conference on Principles of Knowledge Representation and Reasoning*. San Mateo, CA: Morgan Kaufmann Publishers, Inc., 1991. 473~484.
3. Konolige, K., Pollack, M.E. A representationalist theory of intention. In: Bajcsy, R., ed. *Proceedings of the 13th International Joint Conference on Artificial Intelligence*. San Mateo, CA: Morgan Kaufmann Publishers, Inc., 1993. 390~395.
4. Bratman, M.E. *Intentions, Plans and Practical Reason*. Cambridge, MA: Harvard University Press, 1987.
5. Cavedon L, Padgham L, Rao A et al. Revisiting rationality for agents with intentions. In: Xin Yao ed. *Proceedings of the 8th Australian Joint Conference on Artificial Intelligence*. Singapore: World Scientific Publishing Co. Pet. Ltd., 1995.131-138
6. Rao A S, Georgeff M P. Asymmetry thesis and side-effect Problems in linear- time and branching-time intention logic. In: *Proceedings of the 12th International Joint Conference on Artificial Intelligence*. San Mateo, CA: Morgan Kaufmann Publishers, Inc., 1991. 498-504
7. de Vries W, de Boer FS, van der Hoek W, ChMeyer JJ. A truly concurrent model for interacting agents. In: Yuan S-T, Yokoo M, eds. *Intelligent Agents: Specification, Modeling, and Applications: PRIMA 2001*. LNAI 2132, New York: Springer-Verlag, 2001. 16~30.
8. Wang YC, Shi CY. A concurrent BDI-Agent model. *Journal of Software*, 2003,14(3): 422-428
9. Hu SL, Shi CY. An improved twin-subset semantic model for intention of Agent. In Chinese with English Abstract. *Journal of Software*, 2006,17(3):396-402.

# The Communication Model of Migrating Workflow System

Zhaoxia Lu<sup>1</sup>, Dongming Liu<sup>2</sup>, Guangzhou Zeng<sup>1</sup>, and Gongping Yang<sup>1</sup>

<sup>1</sup> School of Computer Science & Technology, Shandong University, Jinan, China 250061  
{luzxia, gzzeng, gpyang}@sdu.edu.cn

<sup>2</sup> School of Materials Science and Engineering, Shandong University, Jinan, China 250061

**Abstract.** Migrating workflow system is one of a practical implementation of mobile agent-based workflow system. To enable effective negotiation and cooperation between two migrating instances of a migrating workflow, communication is an important and essential means. Previous approaches, when applied in the context of mobile agent based workflow, cannot guarantee the reliability and efficiency simultaneously. In this paper, we proposed an improved communication model to solve this problem. The model implements a computation metaphor similar to the way people correspond by mail. It introduces the concept of service domains to imitate postal areas with different postal codes, and decouple mailbox with its owners to ensure reliability. In addition, the combining utilization of Address\_book and friend domain list can locate a target more quickly and exactly. Original experiments show that the model not only ensures reliable communication of moving objects, but also keeps overheads at reasonably low.

## 1 Introduction

Migrating workflow [1,2] is a newly proposed concept to solve the problems coming out when traditional workflow model is used in the inter-organizational environment. It bears the merits of improving the scalability and flexibility of workflow systems with the characteristics of uncertainty and variability. In migrating workflow framework, a business flow is executed concurrently by several migrating instances, each of which is constructed by the concept of mobile agent. The migrating instances performing common business flow form a dynamic alliance. Migrating instances within an alliance must negotiate and cooperate to make decisions and fulfill tasks collectively in virtue of communication with their partners. Therefore, communication scheme among migrating instances is prerequisite to make their collaboration possible. In migrating workflow system, the communication should meet the following requirements.

- **Fault-tolerance and reliability.** The communication scheme should be able to deal with duplicate and out-of-order messages, and solve the key problem of message loss due to the movement of targets.
- **Efficiency and scalability.** A direct message delivery route from the sender to receiver is desirable. In addition, the communication scheme should have the advantage of scalability.

- **Transparent location tracking.** Because entities such as migrating instances are mobile in the network, it is advisable to have mechanisms that allow addressing them in a location-transparent manner.
- **Exactly-once property.** The receiving and disposal of a message have exactly-once property to prevent unnecessary resource consumption.

To our knowledge, though a wide range of communication schemes [3,4,5,6,7,8] have been proposed in mobile agent computing environment, each one of them has its own assumptions, design goals and key problems. None of them meets all of the aforementioned requirements. So when they are applied in the migrating workflow system, they often cause large costs and undesirable restrictions to the system.

In this paper, we present an improved communication model. The model implements a computation metaphor similar to the way people correspond by mail. It divides all working places in the system into several service domains and set up Postoffice for each domain. In addition, some related protocols have been implemented to support reliable and asynchronous communication, efficient location tracking and timely address update.

The rest of this work is organized as follows. Section 2 describes the communication model. The specification of main protocols is discussed in section 3, followed by the analysis and evaluation in section 4. Finally, section 5 provides some concluding remarks and discusses future work.

## 2 Communication Model

The migrating workflow system framework is composed of a number of working places, each of which consists of a docking station server and a working host network. Working places provide running environments and services for the incoming migrating instances. All working places have been classified into several groups called service domains according to the relevancy of services that the working place could provide. Hence each service domain offers a type of services, e.g. query, shopping or delivery, and with one docking station as the manager and coordinator.

A Postoffice is setup for each domain to act as the communication facility for all migrating instances running in it. There are two classes of migrating instances in a domain: local migrating instances (LMI) and active migrating instances (AMI). For a domain  $D$ , a LMI is one created by it, while an AMI is one running in it currently but may be created by another domain.

Let us examine each element of the communication model in details.

**Definition 1 (Postoffice).** A Postoffice is defined by  $(Po\_id, Po\_adr, Mailbox, Address\_book, Post\_agent)$ :

- $Po\_id$  is the identifier of the Postoffice.
- $Po\_adr$  is the address of the docking station that the Postoffice is affiliated to ;
- $Mailbox$  is the set of mailboxes of all migrating instances in the domain  $D_i$  , which comprises two parts:  $Local\_Mailbox$  and  $Active\_Mailbox$ .  $Local\_Mailbox = \{LMB_1, LMB_2, \dots\}$  is the set of mailboxes for indigenes.  $Active\_Mailbox = \{AMB_1, AMB_2, \dots\}$  is the set of mailboxes assigned to active instances.

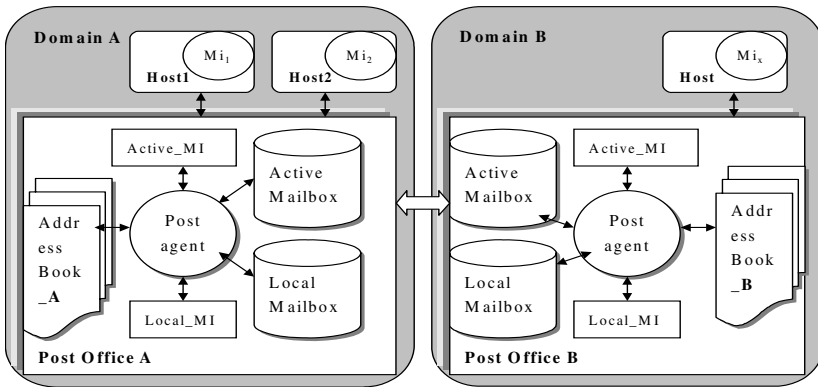
- *Address\_book* = {*ab*<sub>1</sub>, *ab*<sub>2</sub>, ...} records information of acquaintances of the domain *D*<sub>*i*</sub>. An acquaintance is a migrating instance, which has communicated with migrating instances in *D*<sub>*i*</sub>, but does not belong to *D*<sub>*i*</sub>. *Address\_book* is made up of a set of (*MI\_ID*, *CurrentPO*) pairs, where *MI\_ID* is the name of the acquaintance, and *CurrentPO* indicates the Postoffice where the acquaintance currently resides.
- *Post\_agent* administers the two kinds of mailbox and the *Address\_book*, and is in charge of sending, transferring and receiving of messages for migrating instances.

**Definition 2 (Local mailbox, LMB).** A  $LMB \in Local\_Mailbox$  is defined by (*MI\_ID*, *CurrentPO*) :

- *MI\_ID* denotes the name of the corresponding LMI.
- *CurrentPO* logs the postoffice where the LMI currently resides.

**Definition 3 (Active mailbox, AMB).** Each AMB is composed of three elements (*MI\_ID*, *Num*, *Queue*).

- *MI\_ID* is the name of the AMI to which AMB belongs.
- *Num* records the number of messages received when the AMI is running in current domain.
- *Queue* records a list of the received messages. Information about the messages includes the identifier, serial number, title, sender, time and content. The *Post\_agent* may accept or reject a message for a migrating instance according to these information.



**Fig. 1.** The Communication Model

The communication model of migrating workflow is depicted in figure 1. When a migrating instance called *mi* is created in domain A, the *Post\_agent* of A's Postoffice creates a local mailbox (*LMB*) for *mi*. The *LMB* rests on its home Postoffice statically during the lifecycle of *mi* and records all Postoffices that *mi* has passed by. When *mi* is initiated and ready to run, the *Post\_agent* will assign an active mailbox (*AMB*) for *mi*. If *mi* moves to another domain B, its *AMB* in A will be deregistered after it has got a new *AMB* in B. We adopt this method for the sake of reliability rather than let *AMB* move with its owner together.

To achieve the goal of quickly locating a target and lowering the dependency on its home, the historical information about acquaintances are memorized in the Address\_book. Nevertheless the addresses of migrating instances recorded in the Address\_book may be invalid on account of the mobility of migrating instances. To keep the Address\_book updated, we define another concept of friend domain list held by each migrating instance privately to renew the Address\_book timely. Below is the definition of friend domain:

**Definition 4 (Friend Domain).** If  $mi$  is an acquaintance of the domain  $D_i$ , then  $D_i$  is called friend domain of  $mi$ , and  $mi.FriDom = \{D_1, D_2, \dots\}$  describes the friend domain list taken by  $mi$ , where  $D_1, D_2, \dots$  are friend domains of  $mi$ . A domain  $D_i$  satisfying below conditions will be added into  $mi.FriDom$ :

- (1)  $mi \notin D_i.Active\_MI$ . Here  $D_i.Active\_MI$  includes all active migrating instances in domain  $D_i$
- (2)  $mi$  has communicated with at least one migrating instance in  $D_i$ .
- (3)  $D_i \notin mi.FriDom$ .

Then we can conclude that if  $mi$  is in the Address\_book of a domain  $D$ , then  $D$  must be in  $mi.FriDom$ , vice versa.

### 3 Communication Protocols

The communication protocols between two migrating instances include three sub protocols: message sending, message transferring and message receiving.

**Message sending:** Assume a migrating instance  $mi_1$  wants to send message to another one called receiver. To ensure reliable delivery, three cases should be considered: (1) both  $mi_1$  and receiver are running at same domain, then the message can be put into receiver's AMB directly; (2) though  $mi_1$  and receiver are not running at same domain, receiver is an acquaintance of the domain where  $mi_1$  currently reside. We can get the address of receiver from the Address\_book, and send the message to receiver's current domain; (3)  $mi_1$  and receiver are running at different domains, and receiver is not an acquaintance of the domain that  $mi_1$  resides. Since the name of a migrating instance contains information about its home domain, so we can get the address of receiver's home domain by resolving its name. We know that the LMB of the receiver stays at the home Postoffice and records receiver's current domain. So the receiver's current domain address could be acquired by querying its home. Finally, message will be put into receiver's AMB.

**Message transferring:** Messages destined to receiver are put into receiver's AMB. But the delivery might fail if the AMB moves with its owner all together since message might arrive at the time that receiver just leaves for another domain. Let AMB not be deregistered immediately once receiver determines to migrate to another region, but do that after receiver arrives at destination and gets a new AMB. So a message transferring process is necessary for messages delivered to the AMB left behind at from-domain (the domain that receiver left from).

**Message receiving:** Since messages sent to receiver are ultimately put into the AMB at receiver’s current domain, therefore the process of capturing messages is performed completely within the domain. To keep asynchrony and self-determinism of migrating instance, receiver gets its messages in a pull mode [6].

### 4 Analysis and Evaluation

For convenience, some denotations are introduced.  $\delta_{inter}$  and  $\delta_{intra}$  represent the bandwidth expectation of inter and intra domain network, respectively. If a migrating instance A sends a message to B, the probability of A running at the same domain with B is  $\rho_1$ , and  $\rho_2$  is the probability that B is recorded in the Address\_book of the domain where A currently resides. The probability of B being in inter-domain migration state is  $q$ . In addition,  $C_{mes}$  denotes average size of an actual message be, and  $C_{ctl}$  average size of a piece of control message. We define another two terms to describe the communicating and migrating behavior of migrating instances: one is the inter-domain migration ratio  $\eta$ =the total number of inter-domain migration / total number of migration. The other is the spreading degree of its communication partners  $\xi$ = total number of domains involved in communication (i.e. friend domains) / total number of communication.

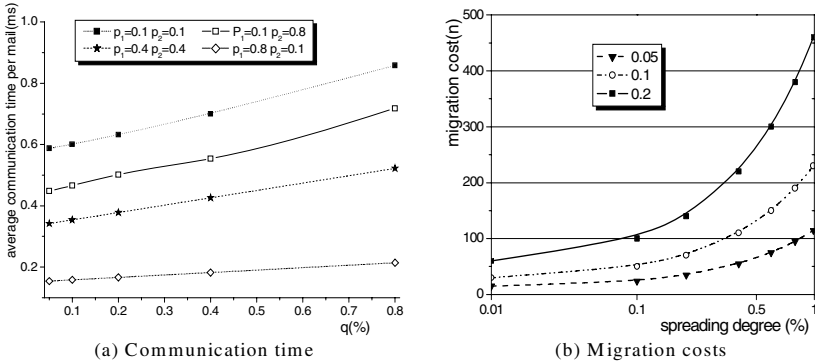


Fig. 2. Experimental Results

In our simulations, the traffic cost per control message ( $C_{ctl}$ ) is 1k, the traffic cost per actual message ( $C_{mes}$ ) is 4k, the bandwidth within a domain is 10 times of that between different domains, i.e.,  $\delta_{intra} = 10\delta_{inter}$ . Figure 2 shows a part of experimental results. Figure 2(a) is communication time overhead under varied  $\rho_1$ ,  $\rho_2$  and  $q$ . The less the  $\rho_1$  and  $\rho_2$ , the more the  $q$ , the longer a communication will be. Figure 2(b) describes migration costs under different migration and communication mode. The total number of migration is 100 for a migrating instance, the inter-domain migration ratios are 0.05, 0.1 and 0.2 respectively, i.e., the instance will perform an inter-domain migration at intervals of 20, 10 and 5 times intra-domain one. The more frequently a migrating instance performs inter-domain migration, the more the migration costs.



Experimental results show that in migrating workflow system,  $\eta$  can be kept between 0.05 and 0.08, while  $\xi$  can be between 0.1 and 0.15. Hence the communication model is very suitable to the migrating workflow system to keep overheads at reasonably low. Moreover, it is also adaptable to frequent communication and intra-region migration of migrating instances.

## 5 Conclusions and Future Works

Compared to previous approaches, our current communication scheme possessed the following advantages: Firstly, the decoupling of mailboxes with its owner could ensure reliable message delivery with exactly-once property. In addition, the communication model can reduce the bandwidth of triangular routing and additional overheads on logon and logout, and can decrease search time and lessen dependency on home. Future works will focus on establishing a more secure communication scheme to be applied in a more general environment.

**Acknowledgment.** This work has been supported by the National Science Foundation of China (grant No.60573169).

## References

1. A.Cichocki, M.Rusinkiewicz, Providing Transactional Properties for Migrating Workflows. *Mobile Networks and Applications* 9,2004,473-480
2. Zeng GZ, Dang Y, The Study of Migrating Workflow Based on the Mobile Computing Paradigm. *Chinese Journal of Computer*, Vol.26, No.10, 2003, pp.1343~1349.(in Chinese)
3. A.L. Murphy and G.P. Picco, "Reliable Communication for Highly Mobile Agents," Proc. 1<sup>st</sup> Int'l Symp. Agent Systems and Applications and 3<sup>rd</sup> Int'l Symp. Mobile Agents(ASA/MA 99), IEEE CS Press, Los Alamitos, Calif., 1999, pp. 141-150.
4. JinHo Ahn. Decentralized Inter-agent Message Forwarding Protocols for Mobile Agent Systems. *ICCSA 2004, LNCS 3045*, pp. 376-385, 2004.
5. Feng XY, Cao JN, Lü J, Chan H, An Efficient Mailbox-Based Algorithm for Message Delivery in Mobile Agent Systems, *LNCS2240*, Springer-Verlag, 2001, p.135-151
6. Jiannong Cao , Xinyu Feng , Jian Lu , Henry C B, Sajal K. Das, Reliable Message Delivery for Mobile Agents: Push or Pull?, *IEEE Transactions on Systems, Man and Cybernetics—Part A: Systems and Humans*, vol. 34, no. 5, September 2004, p.577-587
7. G. Cabri, L. Leonardi, and P. Zambonelli. Mobile-agent coordination models for internet applications. *IEEE Computer*, 33(2):82-89, February 2000.
8. S.Mishra,P.Xie, Interagent Communication and Synchronization Support in the DaAgent Mobile Agent-Based Computing System.*IEEE Transactions on parallel and distributed systems*, Vol.14,No.3,2003,p.290-306

# Multi-user Human Tracking Agent for the Smart Home

Juyeon Lee, Jonghwa Choi, Dongkyoo Shin, and Dongil Shin\*

Department of Computer Science and Engineering, Sejong University, 98 Kunja-Dong,  
Kwangjin-Gu, Seoul, Korea  
{jylee, jhchoi}@gce.sejong.ac.kr, {shindk, dshin}@sejong.ac.kr

**Abstract.** This paper presents a human tracking agent that recognizes the location and motion of the human in the home. We describe the architecture of the human tracking agent, and present an image recognition algorithm to track location and motion of the human. The human tracking agent decides the human's location, which changes in real-time, through the relative distance of home furniture (or appliance) and human. Unlike the human's location, because a person's appearance (height, weight) is different for each person, a human's motion should be recognized to be different from each other person. We converted the image (that is acquired from the network camera) into a standard image (that is defined in the human tracking agent) for recognition of multi-user's motion. We used a LSVM(linear support vector machine) to recognize the feature patterns for human motion. In our experiment, results of motion recognition showed excellent performance accuracy of over 80%.

**Keywords:** User Tracking System, Smart Home, User Motion Recognition.

## 1 Introduction

A home network that integrates sensors, actuators, wireless networks and context-aware middleware will soon become part of our daily life [1]. We define this environment as a smart home [2]. A smart home is a house or living environment that contains the technology to allow devices and systems to be controlled automatically. Also, one of the goals of a smart home is to support and enhance the abilities of its occupants to execute tasks. An artificial intelligence agent in a smart home learns about the occupants and the smart environment, and predicts the appliance services that they will want. That is, it means an intelligent space that provides a suitable service that is predicted from various home contexts that happen in the ubiquitous smart home. The most important contexts among all contexts, which happen in home, are the human's location and motion. Many researchers used various sensor devices (IrDA, ultra audible sound, radio communication signal and the smart floor that analyzes user's location through pressure sensors) [3-9]. But, the various sensors that are presented have the shortcoming that they must attach a communication tag on the user's body. A location recognition method that uses a camera does not attach a communication tag on the human's body, and recognizes the human's motion as well as the human's location from the image offered by the camera. The location recognition system that uses IrDA establishes IrDA sensors in the office, and recognizes the user's location through a signal communication active badge that is

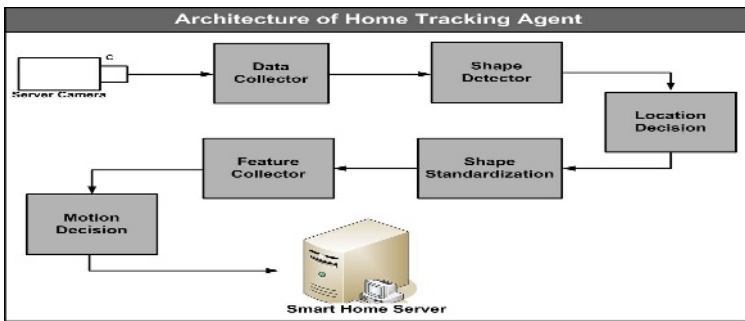
---

\* Corresponding author.

attached to the user [3]. Each active badge has a unique serial number and transmits a serial number to the IrDA sensor that is attached in the office. A location recognition system that uses ultrasonic audible sound has been studied by the University of Cambridge (Active Bat) and MIT (cricket system) [4, 5]. The Active Bat attaches an ultrasonic generator (which is called the Bat) to a person or objects, and attaches an ultrasonic receiver in the office. The cricket that is developed by MIT is a location recognition system based handset. That is, the ultrasonic generator is attached in the office, and the person has an ultrasonic receiver. There is a system named the RADAR developed by Microsoft that is a representative method of the use of RF signals [6]. The Aware Home establishes pressure sensors in the floor, and recognizes user location [7]. The MavHome implemented a user's routing algorithm through use of the pressure sensors in the floor [8, 9].

## 2 Structure of Human Tracking Agent

Figure 1 shows the structure of our home tracking agent.



**Fig. 1.** Architecture of the home tracking agent

The Data Collector module acquires color images (720 X 486) from the digital network cameras every three seconds. The Shape Detector module analyzes the human's volume from four images, which is acquired from the Data Collector module. The Shape Detector module calculates the user's absolute coordinates in home and relative coordinates with reference to the furniture and home appliances. We used a moving window to extract the human's coordinate in each image. The Location Decision module analyses the user's location through coordinates that are offered by the Shape Detector module. The Location Decision module analyses whether the human is near to which home appliance (or furniture) through comparison between the absolute coordinate of human and absolute coordinate of appliance (or furniture) in the home. The Shape Standardization module converts the image (that is acquired by the network camera) into a standard image (that is defined in the human tracking agent) for recognition of multi-user's motion. The Feature Collector module extracts the image that is changed by the Shape Standardization module, and the Motion Decision module predicts the motion of the human.

We applied three kinds of images to acquire absolute coordinates and relative coordinates in the home. The first image is an image in which the user and furniture

(and home appliances) are excluded. The second image is an image in which furniture and home appliances are arranged in the ubiquitous smart home and the third image is an image that includes the human inside the second image. To calculate the human's absolute coordinates in the home, we apply subtraction of three images, and determine the x and y coordinates of the moving window that include the human in the image. When the home tracking agent starts, it executes a subtraction between the first image (which is excluding human, furniture and home appliances) and the second image (which is including furniture or home appliances), and calculates the absolute coordinates of furniture and home appliances using the moving window. The Shape Detector module receives raw image from the Data Collector module, and distinguishes the human's image through subtraction between the second image and third image (that is, including the human with second image), and then calculates the human's absolute coordinates in the home and human's relative coordinates with reference to the furniture and home appliances. It also decides whether a human is near to which furniture (or home appliance). If there is a human in an important place that is defined to the human tracking agent, the Shape Detector module calculates relative coordinates of the moving window, which includes a human, and transmits those to the Location Decision module. We defined the significant places in the home which include sofa, desk, bed, etc. The Location Recognition module can judge a multi-user's location without conversion of the acquired image. Unlike the human's location, because person's appearance (height, weight) differs for each person, the human's motion should be recognized to differ in case of each person. The Shape Standardization module takes charge of motion recognition in the multi-user situation. To recognize the human's motion, we defined a feature sets of six standardized motion. Figure 2 shows six motions that is recognized in the human tracking.

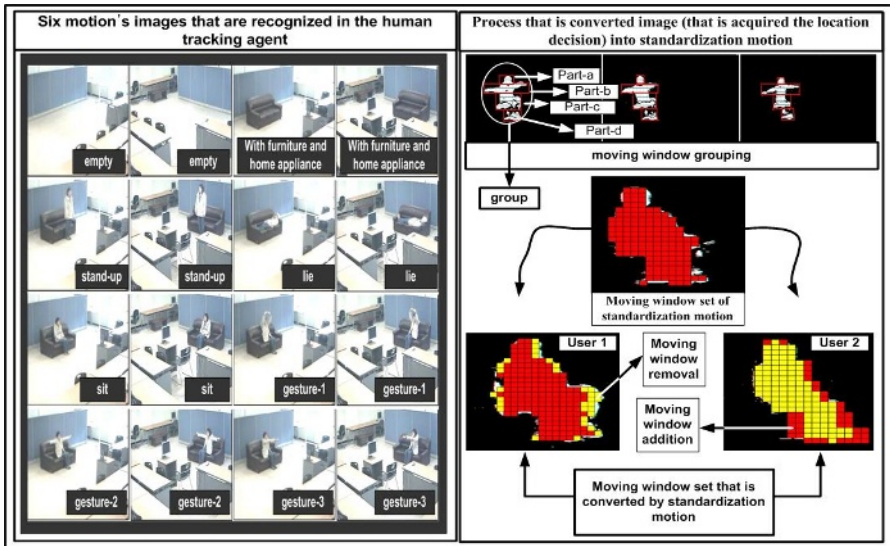


Fig. 2. Six motions that are recognized in the human tracking agent and the process that converts the image (that is acquired the location decision) into standardized motion

The human tracking agent predicts six human's motions (stand-up, lie, sit, gesture-1, gesture-2 and gesture-3). The Shape Standardization module groups moving window that is acquired from the Location Decision module. The algorithm is as follows. The first, count each row's column number and group rows that total column's sum is same (error range is  $\pm 2$ ). Second, add or reduce moving window of each group by comparison between standard motion and motion that is acquired. Third, if count of moving window that is removed (or added) is an even number, the Shape Standardization module removes (or adds) first and last moving window (column) repeatedly, otherwise, removes (or adds) last moving window one more.

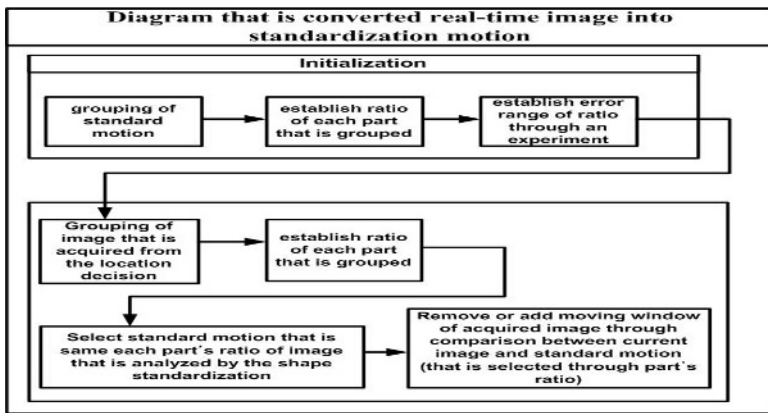


Fig. 3. Diagram that convert real-time image into a standardized motion.

Figure 3 shows diagram that converts real-time image into standardized motion. The feature set that is transmitted from the Shape Standardization module is normalized between 0.1 and 0.9 in the feature collector. And, the Motion Decision module applied normalized features as input values for the LSVM. The Motion Decision module that is presented recognizes the first step motion (sit, lie, stand-up) and second step motion (three gestures).

### 3 Implementation and Evaluations

Our laboratory is 11 m long, 12 m wide and 3 m high. We use network digital cameras to acquire the human's image. The network cameras that are installed in the laboratory include error data, and we overcame this problem through extension of data scope. Nevertheless, the error range of the human's location recognition is 0.024 m. Figure 4 shows subtraction images and the human motion pictures that are acquired in the laboratory.

We experimented with the performance of motion recognition in the human tracking agent, and evaluated performance of the human tracking agent through comparison between single-user and multi-user conditions. Table 1 shows the

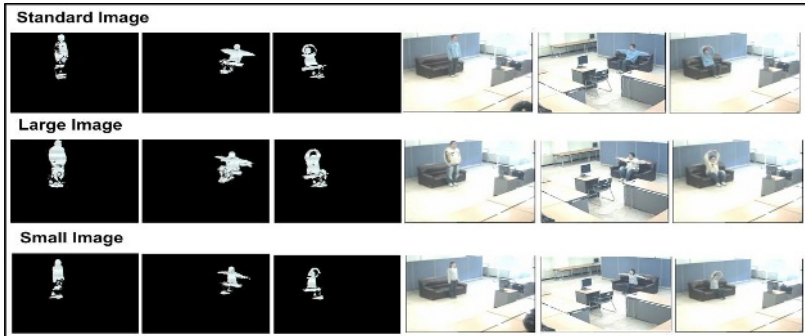


Fig. 4. Pictures of human motion that are acquired in the laboratory

	Number of Support vector	Number of kernel evaluations	Norm of longest vector	Precision on test(%)
Sit	68	13542	2.24578	89
Stand	71	13477	2.32457	91
Lie	64	13356	2.28451	92
Sit_gesture1	65	13487	2.34622	88
Sit_gesture2	68	13587	2.38457	89
Sit_gesture3	69	13257	2.19874	82

Fig. 5. Experimental results with the human tracking agent that recognizes only a single-user

	Number of Support vector	Number of kernel evaluations	Norm of longest vector	Precision on test(%)
Sit	57	12992	2.21434	81
Stand	65	13023	2.28132	84
Lie	54	13112	2.27878	85
Sit_gesture1	59	12843	2.31436	79
Sit_gesture2	61	12934	2.39105	80
Sit_gesture3	61	13998	2.21769	76

Fig. 6. Experimental results for the human tracking agent that recognizes multiple users

performance of motion recognition by the human tracking agent that is limited to single-user operation. The six kinds of motion that are recognized in the human tracking agent are shown in Figure 5 to have a high average accuracy rate of 88.5%. The sit\_gesture3 shows a lower accuracy rate of 82% because the feature set of the sit\_gesture3 does not have a feature that is clearly distinguishable from other motions.

Figure 6 shows the performance of the human tracking agent that recognized multiple users. The human tracking agent decreases performance of motion recognition in multi-user mode in comparison with recognition in single-user mode. But, still, results of motion recognition showed an excellent performance of over 80%.

We are currently studying effective motion recognition system by using the feature's relative importance.

## 4 Conclusions

This paper presents a human tracking agent that recognizes the location and motion of the human in the home. The human tracking agent decides the human's location, which changes in real-time, through the relative distance of home furniture (or appliance) and human. Unlike the human's location, because a person's appearance (height, weight) is different for each person, a human's motion should be recognized to be different from each other person. We converted the image (that is acquired from the network camera) into a standard image (that is defined in the human tracking agent) for recognition of multi-user's motion. The human tracking agent that is presented in this paper support multi-user motion recognition. We used a LSVM to recognize the feature patterns for human motion. In our experiment, results of motion recognition showed excellent performance accuracy of over 80%.

## References

1. Schulzrinne, H, Xiaotao Wu, Sidiroglou, S, Berger. S.: Ubiquitous computing in home networks. *Communication Magazine. IEEE*, Vol. 41, Issue. 11, (2003) 128-135
2. Sherif, M. H.: Intelligent homes: a new challenge in telecommunications standardization. *Communication Magazine. IEEE*, Vol. 40, Issue.1, (2002) 8-8
3. Want, R., Hopper, A.: Active badges and personal interactive computing objects. *Consumer Electronics. IEEE Transactions on* Vol. 38, Issue. 1, (1992) 10 - 20
4. Hazas, M.; Hopper, A.: Broadband Ultrasonic Location Systems for Improved Indoor Positioning. *Mobile Computing. IEEE Transactions on* Vol. 5, Issue. 5, (2006) 536 - 547
5. Adam Smith, Hari Balakrishnan, Michel Goraczko, Nissanka Bodhi Priyantha.: Tracking Moving Devices with the Cricket Location System. 2nd International Conference on Mobile Systems. Applications and Services (Mobisys 2004), (2004)
6. p.Bahl and V.Padmanabhan.: RADAR: An In-Building RF-based User Location and tracking system. *Proc, IEEE infocom, IEEE CS Press, Los Alamitos, CA, (2000) 775-784*
7. Orr, J. Robert, D. Gregory, Abowd.: The smart floor: A Mechanism for natural user identification and tracking. To appear in the proceedings of the 2000 Conference on Human Factors in Computing systems (CHI 2000), The Hague, Netherlands, (2000)
8. MavHome, <http://cygnus.uta.edu/mavhome/>
9. Das. S.K, Cook, D.J, Battacharya. A, Heierman. E.O. III, Tze-Yun Lin.: The role of prediction algorithms in the MavHome smart home architecture. *Wireless Communications. IEEE*, Vol. 9, Issue. 6, (2002) 77 - 84

# Towards Embedding Evolution into a Multi-agent Environment

Chantelle S. Ferreira and Elizabeth M. Ehlers

Academy for Information Technology, University of Johannesburg, Auckland Park  
Kingsway Campus, P.O. Box 524, Auckland Park, Johannesburg 2006, South Africa  
920101308@student.uj.ac.za, eme@rau.ac.za

**Abstract.** Due to the evolutionary nature of humans, systems are required to be continuously replaced because of their inability to meet or adapt to the changing needs of humans. Such replacements are costly and time consuming. Therefore this paper proposes a cheaper and quicker alternative, which is to embed self-evolving mechanisms into a multi-agent environment, thus giving the agents in the environment the ability to either meet or exceed the way in which humans evolve.

## 1 Introduction

The combined works of Charles Darwin and G.J Mendel stated that all species were created through a process known as evolution and variations are passed from one generation to the next through genetics. The heart of Darwin's theory on evolution is based on natural selection. Individuals, which contain a variation that enables them to survive, reproduce and therefore evolve into superior species [1, 2, 3].

Artificial evolution is a field of study which merges inspiration from biology with the tools and goals of computer science and artificial intelligence [4]. The following section will be devoted to understanding the capabilities of current multi-agent environments.

## 2 Multi-agent Environment

In a multi-agent environment, conflict between two agents, a phenomenon which emerges autonomously, can either be resolved in a competitive or cooperative manner [5]. Cooperation between agents make it possible for a collection of simple agents to act in an apparently intelligent manner [6, 7, 8]. Competition between agents ensures that continuous adaptations in some agents will force adaptations in other agents [4, 9].

Single agents currently require the capability to withstand and adapt to different sources of anticipated and unanticipated environmental changes during run-time. The environmental changes can include [5, 10, 11, 12, 13]: resource variability; intentional disruptions and intrusions; changing user needs cascading failures; and system faults. Evolutionary agents are required because single agents in a multi-agent environment do not meet all the requirements of a changing environment and tend to act poorly when change is introduced.



### 3 Embedded Evolution Model

An agent can be completely rewritten to include evolution [14], but programming every agent to include evolution is very cumbersome. The benefits for embedding evolution into an existing agent are that the agent contains domain knowledge; current goals that are applicable to the current environment; and information on the sensor and actuator capabilities.

Such benefits relieve the programmer of programming domain and hardware specific information into each evolutionary agent, allowing the creation of a generically developed evolutionary piece of code. An agent is embedded with evolution by inserting a separate evolutionary piece of code between the agent and its environment. The evolutionary piece of code is required to obtain information from the unknown agent and manage the relationship between the unknown agent and its environment; and to evolve to meet the changing environment needs. The inference engine of the evolutionary agent will be the focus of the discussion on the embedded evolutionary model.

#### 3.1 Inference Engine

The inference engine meta-management process requires information on the environment, knowledge base and agent's goal. The meta-management process is required to coordinate between the deliberative and the reactive process; if the meta-management requires more information to coordinate, the evolutionary agent is required to explore the environment. This coordination is accomplished with the use of alarms that detects urgent, important and critical conditions. The meta-management process gives the deliberative processes control over the reactive process until an alarm is triggered. The deliberative process regains control once the reactive process has handled the situation. The reactive process simply consists of condition action rules, the condition-action rules are required by the evolutionary agent to handle failures which require quick response and little cognitive processing. The deliberative process is the more important part of the intentions component and more complex than the reactive process. The deliberative process is discussed in the subsequent section. [8, 13]

##### 3.1.1 Deliberative Process

The greatest benefit required from an evolutionary agent is a comprehensive adaptation methodology that spans adaptation from small to large systems and supports the entire range of adaptations [10]. The deliberative process possesses the entire internal symbolic reasoning model required by any evolutionary agent [14]. Figure 1 demonstrates the evolutionary agent's deliberative process.

Information received by the deliberative process from the environment is analyzed by the critic function to determine the agent's success. The agent's success with the corresponding action is saved to the knowledge base for future use by the evolutionary agent. The deliberative process can also use the information analyzed by the critic function to determine what the evolutionary agent has learnt; the learnt information is used to update the reactive process' condition-action rules, observation planning component and the adaptation planning component. To handle failures the

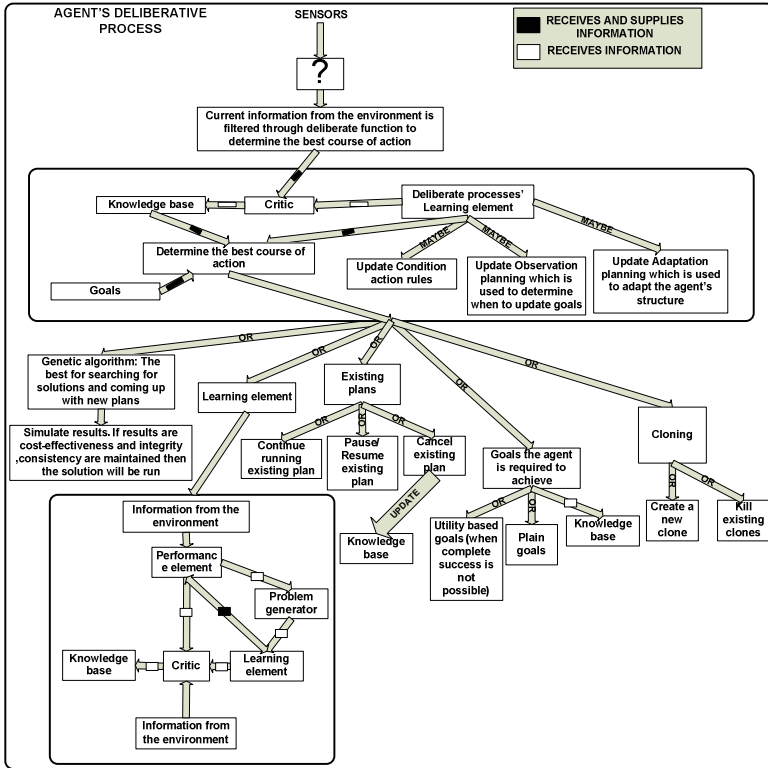


Fig. 1. The evolutionary agent's deliberative process

condition-action rules are required to be updated to include what the agent has learnt. The observation planning component is responsible for determining which goals the evolutionary agent requires by observing the external environment and the agent internally. The goals can include adaptations. The observation planning component is updated to include the state of the current environment and results of previous goals and non goal acceptance decisions. The adaptation planning component determines exactly which adaptations from the observation planning component to implement and when. The adaptation planning component is updated to include the results of previous adaptations and lack of adaptations. Once the deliberative process has finished updating, the deliberative process determines the evolutionary agent's next action. The deliberative function uses information from the knowledge base, current agent goals and information learnt to determine which process the agent should use in determining the action to be implemented. The processes include: genetic algorithms; learning element; existing plans; goals the agent is required to achieve and cloning.

3.1.1.1 *Agent's Adaptation.* The evolutionary agent's plans can include adaptations to the evolutionary agent. Adaptability is required by the evolutionary agent to respond to a dynamic environment. Evolutionary agents are required to adapt to changes in the

operating environment or to the agent internally in real-time. The adaptability process gives the evolutionary agent the ability to adapt to both static and dynamic aspects of the agent. The static aspects of the agent include the agent's structure and topology; and the dynamic aspects of the agent include the agent's behavioural and interaction. Figure 2 demonstrates how the evolutionary agent adapts. [11]

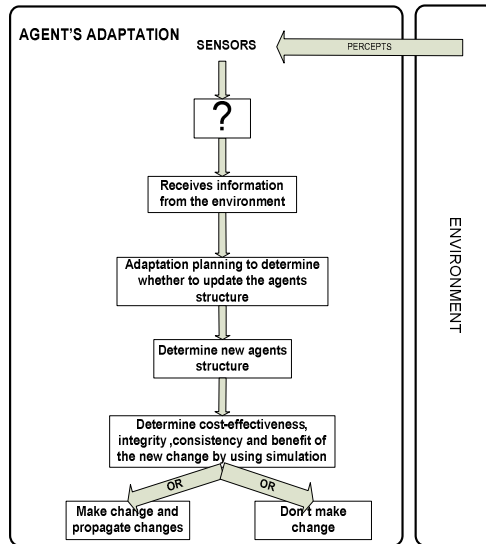


Fig. 2. The evolutionary agent's adaptation process

## 4 Implementation

An evolutionary agent puzzle solver was developed to demonstrate the feasibility and problem-solving accuracy of the model discussed above. In addition to accuracy being an important factor in the success of completing a puzzle, the time required to solve a given puzzle grows exponentially to the number of puzzle pieces involved.

The evolutionary agent puzzle solver attempts to address both concerns of accuracy and time by reaching a compromise between the two requirements when attempting to solve a puzzle problem. To test the viability of the model discussed, an evolutionary agent puzzle solver was embedded into a group of puzzle solvers that use existing algorithms such as the greedy best first search (GBFS), GBFS modified and uniform search.

The evolutionary agent puzzle solver has the ability to learn which algorithm in the group of algorithms best solves a puzzle, based on the parameters specified by a human user. The parameters include minimum and maximum time; and minimum and maximum accuracy and whether time or accuracy is more important. The evolutionary agent puzzle solver also creates its own algorithm with parameters specified

from a genetic algorithm. The success of the genetic algorithm is determined by its user, as the human user assigns fitness values to the genetic algorithm. The evolutionary agent’s algorithm solves a puzzle by splitting the puzzle in half, first solving the first half and finally completing the puzzle; the puzzle is split using the parameters obtained from the genetic algorithm.

The evolutionary agent’s learning component enables a user with no knowledge of the program to determine the best puzzle solver to solve a puzzle that meets the user’s specified requirements. The evolutionary agent’s algorithm performance improves considerably as the size of the puzzle increases, proving the implementation of the proposed model not only as possible, but outperforming existing algorithms. Table 1 contains results of various puzzles solved by the different algorithms. The values in the table vary based on the puzzle to be solved.

**Table 1.** Comparison of results as generated by the evolutionary algorithm against other existing algorithms

Algorithm	Size of puzzle			Time in seconds			Steps			Accuracy		
	4	9	64	0	2	40	67395	443605	34231629	100	100	100
GBFS	4	9	64	0	2	42	67655	443792	39646783	100	100	100
GBFS modified	4	9	64	0	2055	?	53064	1508861151	?	100	20	?
Uniform Search	4	9	64	0	2055	?	53064	1508861151	?	100	20	?
Evolutionary algorithm	4	9	64	1	2	29	214359	689362	22541992	100	100	100

## 5 Future Work

More in-depth research is required on computer ethics and agent interoperability. Required computer ethics research also includes the enforcement of ethics on all computer systems, especially as systems become more autonomous. Required agent interoperability research also includes developing standards and supporting infrastructure in the development of all computer agents [15].

## 6 Conclusion

The agents in a multi-agent environment currently do not meet all the requirements of a changing environment and therefore tend to act poorly when change is introduced. Embedding evolution in agents attempt to construct evolutionary agents that exhibit aspects which allow them to evolve in a changing environment. It is also difficult to ensure that an evolutionary agent will evolve desirable behaviour, especially when multiple dimensions of the applications’ overall success exists [12]. Since several components of the evolutionary agent will be obtained from the agent to be embedded, a structured path will be created for the evolutionary agent to follow [16].

## References

1. The New Encyclopaedia Britannica. 15th edn. Volume 4. Encyclopaedia Britannica, Chicago (1986) 622-623
2. Lenski, R.E., Ofria, C., Pennock, R.T., Adami, C.: The evolutionary origin of complex features. *Nature*. Volume 423(6936), (2003) 139-144
3. Taylor, T.: On self-reproduction and evolvability. *ECAL* (1999) 94-103
4. Wiegand, R.P.: An analysis of cooperative coevolutionary algorithms. PhD thesis, George Mason University, 2003
5. Floreano, D., Urzelai, J.: Artificial Evolution of Adaptive Software: An Application to Autonomous Robots. *3D The Journal of Three dimensional images*. Volume 14(4), (2000) 64-69
6. Maes, P.: Intelligent Software. *Scientific American*. Volume 273(3), (1995) 84-86
7. Rodriguez-Fortiz, M. J., Paderewski-Rodreguez, P., Garcia-Cabrera, L., Parets-Llorca, J.: Evolutionary modelling of software systems: its application to agent-based and hypermedia systems. In *IWPSE '01: Proceedings of the 4th International Workshop on Principles of Software Evolution*, (2001) 62-69
8. Sloman, A., Scheutz, M.: A framework for comparing agent architectures. In *UKCI '02: UK Workshop on Computational Intelligence*, Birmingham. 2002
9. Guessoum, Z., Rejeb, L., Durand, R.: Using adaptive multi-agent systems to simulate economic models. In *AAMAS'04: Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems*, (2004) 68-75
10. Oreizy, P., Gorlick, M.M., Taylor, R.N., Heimbigner, D., Johnson, G., Medvidovic, N., Quilici, A., Rosenblum, D.S., Wolf, A.L.: An Architecture-Based Approach to Self-Adaptive Software. *IEEE Intelligent Systems*. Volume 14(3), (1999) 54 - 62
11. Dashofy, E.M., Hoek, A.V.D., Taylor, R.N.: Towards architecture-based self-healing systems. In *WOSS '02: Proceedings of the first workshop on Self-healing systems*, (2002) 21-26
12. Helsinger, A., Kleinmann, K., Brinn, M.: A framework to control emergent survivability of multi agent systems. In *AAMAS '04: Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems*, (2004) 28-35
13. Amin, M.: Toward Self-Healing Infrastructure Systems. *Computer*. Volume 33(8), (2000) 44-53, IEEE Computer Society Press
14. Nwana, H.S., Ndumu, D.T.: An introduction to agent technology. In *Software Agents and Soft Computing*, (1997) 3-26
15. Bradshaw, J.M.: KAoS: An open agent architecture supporting reuse, interoperability, and extensibility. In *Tenth Knowledge Acquisition for Knowledge-Based Systems Workshop*, 1996
16. Foster, I., Jennings, N.R., Kesselman, C.: Brain meets brawn: why grid and agents need each other. In *AAMAS'04: Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems*, (2004) 8-15

# A Multi-agent Framework for Collaborative Product Design

Jian Xun Wang and Ming Xi Tang

School of Design, The Hong Kong Polytechnic University  
Hung Hom, Kowloon, Hong Kong, China  
{Jianxun.wang, sdtang}@Polyu.edu.hk

**Abstract.** To survive the increasing worldwide competition, new technologies are required by product manufacturers to improve both effectiveness and efficiency of collaborative design and. In this paper, a computational model of collaborative product design is proposed to facilitate the management and coordination of the collaborative product design process. Based on this model, the system architecture of a multi-agent framework for collaborative design is discussed. A software prototype featured by flexibility, scalability, distributed decision-making, and etc., is now being developed.

## 1 Introduction

A complex product design task usually requires the close cooperation among a number of multidisciplinary designers. On one hand, nowadays products are getting more and more complicated; on the other hand, increasing worldwide competition and rapidly changing customers' demands are forcing manufacturers to produce high quality products with shortened time-to-market. Thus, new technologies are required by product manufacturers to improve both effectiveness and efficiency of collaborative design and sharpen their competitiveness.

Collaborative design can create added value in the design and production process by bringing the benefit of team work and cooperation in a concurrent and coordinated manner. However, distributed design knowledge and product data make the design process cumbersome. Furthermore, the difficulties arising from the differences between heterogeneous system architectures and information structures undermine the effectiveness and the success of collaborative design.

This paper presents an ongoing project on the application of intelligent software agents to collaborative product design. In this project, a computational model of collaborative product design is proposed to facilitate the management and coordination of the collaborative product design process. Based on this model, the system architecture of a multi-agent framework for collaborative design is discussed. A software prototype featured by flexibility, scalability, distributed decision-making, and etc., is now being developed.

## 2 Related Works

As an emergent approach to developing distributed systems, agents have mostly been used for supporting cooperation among designers, providing a semantic glue between traditional tools, or for allowing better simulations [1]. One of the earliest projects in this area is PACT which demonstrates the use of agents to combine pre-existing engineering systems to constitute a common framework, using facilitators and wrappers, by adopting a federated architecture [2]. DIDE is developed to achieve collaboration and openness in an engineering design environment through the integration of CAD/CAM tools, databases, knowledge base systems, and etc [3]. SHARE project conducted at Stanford University, USA, including Next-Link [4, 5], and Process-Link [6], aimed at using agents to help multidisciplinary design engineers track and coordinate their design decisions with each other in the concurrent design of aircraft cable harnesses with the support of a Redux agent. Liu and Tang [7] presented a multi-agent design environment that supported cooperative and evolutionary design.

Although agent technology has been considered very promising for developing collaborative design systems, and most of the systems that have been implemented so far are domain dependent, and were intended for integrating legacy design tools. Such systems are still at a proof-of-the-concept prototype development stage. Furthermore, only a few literatures mentioned design process model for supporting dynamic design project management in a multi-agent collaborative design system. A detailed discussion on the issues and challenges in developing multi-agent design systems can be found in literature [8].

In this paper, we present a computational model of collaborative design project management and which forms the core of a multi-agent framework for collaborative product design.

## 3 The Approach

A product design project often requires the cooperation among designers or design teams who are usually working on different locations and at different times. Usually, such a project can often last for a long period of time, during which the product requirements may have to be changed. Also, during the design process, various decisions are made by each individual participating in the design project. These decisions have consequences and impose additional constraints to others involved in the design process. It is very important to coordinate the whole design process effectively so that the overall product design process progresses timely and accurately. To address these management problems, a novel computational model for collaborative product design management is proposed (Fig. 1). Four types of main roles including Design Project Manager, Design Agent, Product Data Manager, and Constraint Manager, are identified in a collaborative design process as followed.

Next, we will illustrate how this computational model works to facilitate collaborative product design.

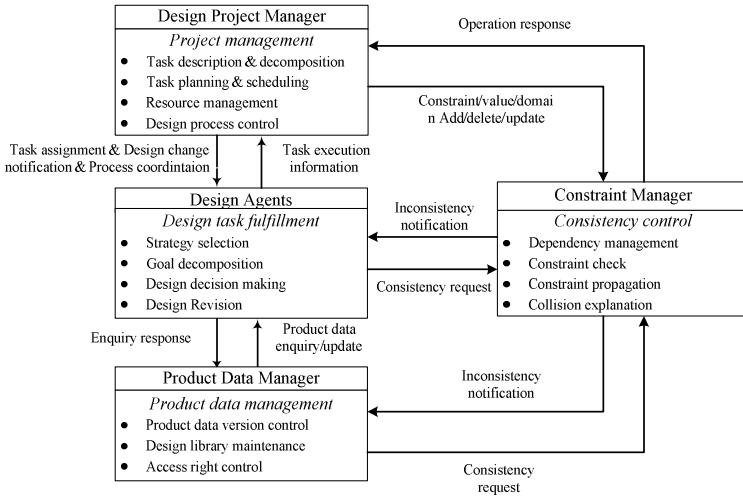


Fig. 1. A computation model of the collaborative product design management

For a new product design project, the first step is to identify the specification of the design task, i.e. the design goal, which the final design output will be evaluated against. A design task is associated with a set of attributes including inputs, expected outputs, duration, start time, end time, and assigned agents. Inputs and outputs are passed in the form of product part/assembly, parameters, which can represent data as well as documents or physical objects. A task can be further decomposed into several subtasks. In this way, the decomposition of the overall design project can be viewed as a design task tree. Then, the design tasks can be planned by defining the logical links between inter-related tasks and be scheduled by specifying the resources as well as start and end times to achieve the design plan. Design management works mentioned above are all carried out by the Design Project Manager role.

After the planning and scheduling of design tasks, Design Agents will receive design task assignments. According to the design task specification and attributes, Each Design Agent adopts some specific strategy and makes design decisions to fulfill the assigned design task by itself or cooperating with other design agents.

During the design process, the Product Data Manager is responsible for product data version control, design library maintenance, and access right control. The Constraint Manager takes charge of the design dependency management. When there is a design decision made by one Design Agent, a constraint check event is triggered to examine whether any design constraint is violated. If any constraint is not consistent, involved Design Agents are informed of the constraint violation with a collision explanation so that participants can understand the rationales behind design decisions.

The Design Project Manager is also responsible for supervising the execution of the scheduled tasks. When a new task occurs, or the input for one of a design agent's tasks becomes available or changed, or a task or a task assignment is invalidated, notifications will be sent to the concerned design agents with the support of constraint manager.



## 4 Implementation of the Multi-agent Framework

Based on the computational model of product design management described in previous section, we propose an agent-based system to facilitate, rather than automate, teamwork in a collaborative product design project. It is organized as a network of intelligent agents which interact with each other and participating team members to facilitate collaborative design work (Fig. 2). These agents are briefly described below.

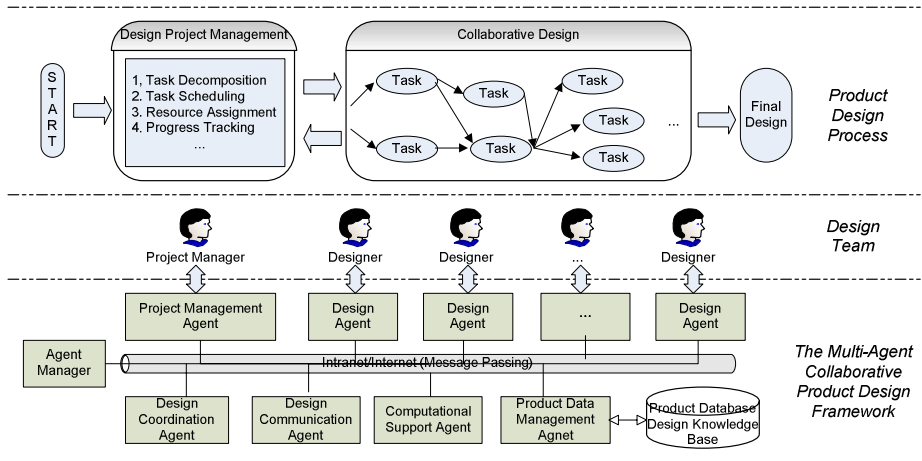


Fig. 2. General architecture of the agent-based collaborative product design system

- Project Management Agent** helps project manager to describe the design project, specify the initial parameters and design constraints, decompose a complex design project into subtasks, assign them to designers, schedule them and track the whole design process. It has a user interface for assisting the project manager in managing the project plan and schedule, keeping track of progress of the project, and more importantly, making necessary changes while the plan gets more detailed and the higher levels of the plan have to be updated. Also, it serves to remind the project manager and involved designers of the outstanding issues related to schedule execution.
- Design Agents** encapsulate some traditional popular CAD systems and are used by designers to fulfill design tasks through cooperation with other agents. There have been some standards, such as CORBA, DOM, allowing for legacy applications to be encapsulated using wrapper classes and to behave as distributed components.
- Product Data Management Agent** is responsible for managing the product database and ensuring the consistency of the product data in the product design process, and informing concerned design agents of the product data change event (such as submission, modification and so on) made by other design agents.
- Design Coordination Agent** serves to coordinate the design process through design constraint propagation, notifying involved designers of constraint conflicts and their respective reason, and helping designers address the conflicts.

- **Design Communication Agent** provides support for interaction among designers by services, such as email, video/audio conferencing, file transfer service, application sharing, whiteboard, and etc or coordination message transporting service among other agents and human designers.
- **Computational Support Agent** accounts for engineering calculations or analysis upon request. It may be a Finite Element Analysis tool, Optimizer of some other engineering computation tools.
- **Agent Manager** serves as the runtime environment for all other agents and it is responsible for controlling the utilization and availability of all agents by maintaining an accurate, complete and timely list of all active agents through which agents residing in the system are capable of cooperating with each other.

All the software agents are connected by a local network (LAN) via which they communicate with each other. Also, the external design agents residing on the internet carrying out specific design tasks can communicate with agents located in the local network via the Internet.

Our system adopts a Collaborative Product Data Model (CPDM) which is established by extending traditional parametric feature-based model to encompass collaborative design management and coordination information. In this way, the dynamic product design evolving process can be captured and a design constraint network could be maintained. Based on the CPDM, our system employs a constraint-based Collaborative Design Process Model (CDPM) which views a collaborative design process as a sequence of state transitions from one design state to another resulting from the design decisions made by designers and the precondition of design state transition is the consistency of all design constraints being preserved. A more detailed description of the CPDM and the CDPM can be found in authors' other papers [9, 10].

Our agent-based system is now being implemented on a network of PCs with Windows 2000/XP and Linux operating systems. Java is chosen as the primary programming language for the system implementation. C++ is also used as the programming language for the integration of legacy systems. JNI (Java Native Interface) serves as the glue between Java-written applications and C++-wrapped native applications. JADE (Java Agent DEvelopment Framework) which is a software framework fully implemented in Java language is utilised to develop the agent-based system. Currently, Inventor<sup>®</sup>, as one of the most popular CAD systems, is being encapsulated into our design agents through Inventor<sup>®</sup> COM API. FIPA ACL serves as the agent communication language. MySQL<sup>™</sup> is used by the product data management agent as the database system for storing product data and design knowledge. The entire prototype of the system will be implemented before mid 2006.

## 5 Conclusions

In this paper, on the basis of the problem identification and the reviews of the research in relevant areas, a computational model of collaborative product design management is proposed firstly. Then, a multi-agent framework supporting collaborative product design to facilitate the management and coordination of collaborative product design

process via the cooperation of a network of intelligent agents is proposed. The system architecture of our proposed system is presented. A software prototype featured by flexibility, scalability, distributed decision-making, and etc., is now being developed. More experiments are required to be carried out in order to test and improve our system. In the future testing of our approach and software, we intend to involve designers in the process with real design examples since we believe that a design-oriented approach needs to be taken in order to identify those key tasks that need collaboration and support by software agents.

## Acknowledgements

This research project is supported by a UGC PhD project grant from the Hong Kong Polytechnic University.

## References

1. Shen, W. and Wang, L.: Web-Based and Agent-Based Approaches for Collaborative Product Design: an Overview, *International Journal of Computer Applications in Technology*, Vol. 16(2/3) (2003) 103–112.
2. Cutkosky, M.R., Engelmores, R.S., Fikes, R.E., Genesereth, M.R., Gruber, T.R., Mark, W.S., Tenenbaum, J.M., and Weber, J.C.: PACT: An experiment in integrating concurrent engineering systems, *IEEE Computer*, Vol. 26(1) (1993) 28–37.
3. Shen, W., Barthes, J.P.: An experimental environment for exchanging engineering design knowledge by cognitive agents, In Mantyla, M., Finger S., and Tomiyama T. (Eds.): *Knowledge intensive CAD-2*, Chapman and Hall (1997) 19–38.
4. Park, H., Cutkosky, M., Conru, A., Lee, S.H.: An Agent-Based Approach to Concurrent Cable Harness Design, *Artificial Intelligence for Engineering Design, Analysis and Manufacturing*, Vol. 8(1) (1994) 45–62.
5. Petrie, C., Cutkosky, M., Webster, T., Conru, A., Park, H.: Next-Link: An Experiment in Coordination of Distributed Agents, *AID-94 Workshop on Conflict Resolution*, Lausanne (1994).
6. Goldmann, S.: Procura: A Project Management Model of Concurrent Planning and Design, *Proceeding of WET ICE'96*, Stanford, CA (1996).
7. Liu, H., Tang, M.X., Frazer, J.H.: Supporting evolution in a multi-agent cooperative design environment, *Advances in Engineering Software*, Vol. 33 (6) (2002) 319–328.
8. Lander, S.E.: Issues in Multiagent Design Systems, *IEEE Expert*, Vol. 12(2) (1997) 18–26.
9. Wang, J.X., Tang M.X.: Knowledge Representation in an Agent-Based Collaborative Product Design Environment, *Proceedings of the 9th International Conference on Computer Supported Cooperative Work in Design (CSCWD 2005)*, Vol. 1, IEEE Computer Society Press, (2005) 423–428.
10. Wang, J.X., Tang M.X.: An Agent-Based Approach to Collaborative Product Design, *Proceedings of IDETC/CIE 2006, ASME 2006 International Design Engineering Technical Conferences & Computers and Information in Engineering Conference*, Philadelphia, Pennsylvania, USA, (2006) DETC2006-99149.

# Research on Algorithms of Gabor Wavelet Neural Network Based on Parallel Structure

Tingfa Xu, Zefeng Nie, Jianmin Yao, and Guoqiang Ni

School of Information Science and Technology, Laboratory of Photoelectric Imaging and Information Engineering, Beijing Institute of Technology, Beijing 100081, China

**Abstract.** Aiming at image target recognition, a novel algorithm of Gabor wavelet neural network based on parallel structure is proposed in this paper, and the system of neural network for multi-target recognition is designed. Based on the characteristics of multi-CPU parallel structure and the parallel property of neural network, the algorithm of Gabor wavelet neural network is proved theoretically. The relevant algorithm structure is designed; the training and recognizing algorithms for image target recognition are given out. Finally, the simulation experiment for 4 types of plane targets indicated that recognition rate reached 98%+, recognizing time was 40ms.

## 1 Introduction

Since the conception and algorithm of transform network was presented by Qinghua Zhang and Benveniste in 1992 [1], there have been various models of wavelet neural network. Such as discrete affine transform neural network by Pan and Krishnaprasad [2], orthonormal multi-resolution transform neural network by Stephanopoulou [3], orthonormal transform base neural network based on compactly supported scaling functions by Jun Zhang etc [4], Gabor atom neural networks with application in radar target recognition by Yu Shi and Xianda Zhang [5], and so on. It has been proved that wavelet neural network is asymptotically optimal approximator for functions of one variable [6].

Aiming at image target recognition, a novel algorithm of Gabor wavelet neural network based on parallel structure is presented in this paper, and the system of neural network for multi-target recognition is designed. Mainly based on the characteristics of multi-CPU parallel structure and the parallel nature/property of neural network, Gabor wavelet neural network algorithms are designed combined with Gabor transform and improved BP neural network algorithms. The relative algorithm structure is constructed, along with the training algorithm for gray image target recognition.

## 2 Gabor Wavelet Neural Network Algorithm

### 2.1 Gabor Wavelet Volume

Definition 1: for a given signal  $f(t)$ , the expanded formula with exponential basal function  $g_p(t)$  modulated by Gauss function can be defined as

$$f(t) = \sum_{p=1}^{\infty} B_p g_p(t). \tag{1}$$

$$g_p(t) = (\pi\sigma_p^2)^{-0.25} \exp\left[-\frac{(t-t_p)^2}{2\sigma_p^2}\right] \exp(j2\pi f_p t). \tag{2}$$

where exponential function  $g_p(t)$  has an adjustable variance  $\sigma_p^2$  and an adjustable temporal and frequency center  $(t_p, f_p)$ .

Definition 2: Gabor Wavelet volume can be defined as

$$g_{u,\xi,s}(t) = \frac{1}{\sqrt{s}} g\left(\frac{t-u}{s}\right) e^{j\xi t}. \tag{3}$$

where  $g(t) = \sqrt[4]{2}e^{-t^2}$  is the basal function antetype of Gabor Wavelet volume, whose temporal offset, frequency offset and scaling variance form different basal functions. And where  $s > 0, t_p = u$  and  $\xi = 2\pi f_p$ .

### 2.2 Structure of Gabor Wavelet Neural Network

The structure of Gabor wavelet neural network with 3-D Gabor Transform for target classification, is a typical multilayer feed-forward neural network, as shown in fig. 1. Aiming at classifying N groups images into P types, the input of network is  $\{x_i, i = 1, 2, \dots, N\}$ , and the output is  $y_m, m = 1, 2, \dots, M$ , which decides the target types. Take  $M = 4$  for example,  $y_1 y_2 y_3 y_4 = 1000, 0100, 0010, 0001$  indicates that the output is 1, 2, 3, 4 types targets.

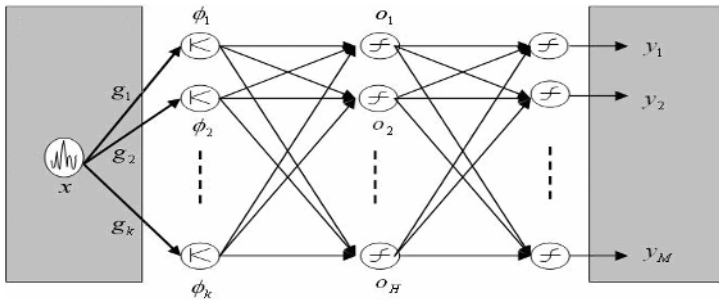


Fig. 1. The structure of Gabor wavelet neural network

Suppose that there are  $L$  data in every group observation signal, namely  $x_i = [x_i(1), x_i(2), \dots, x_i(L)]^T$ , let  $\{g_k, k = 1, 2, \dots, K\}$  denotes Gabor basal function vector with  $K$  kinds of scaling, where  $g_k = [g_k(1), g_k(2), \dots, g_k(L)]^T$ , which makes weighted summation of  $L$  data of target vector  $x_i$ , and  $g_k(t)$  is defined as

$$g_k(t) = \frac{1}{\sqrt{s_k}} g\left(\frac{t - u_k}{s_k}\right) e^{j\xi_k t}, k = 1, 2, \dots, K. \tag{4}$$

which  $s_k$ 、 $u_k$ 、 $\xi_k$  separately denote the scaling, offset and frequency modulation parameter.

These parameters adaptably adjust to make certain cost function minimization.

The structure of Gabor wavelet neural network as shown in Fig. 1 is:

(1) Input layer: There are  $K$  nodes  $\phi_1, \phi_2, \dots, \phi_k$  in this layer, where  $\phi_{ik}$  denotes the  $K_{th}$  node of the Gabor basal function related to  $i$  group target signal vector. Hence,  $\phi_{ik}$  can be defined as the absolute value of inner-product of Gabor basal function vector  $g_k = g(u_k, \xi_k, s_k)$  and target vector  $x_i$ , namely

$$\phi_{ik} = \left| \langle g_k, x_i \rangle \right| = \left| \int \frac{1}{\sqrt{s_k}} g^* \left( \frac{t - u_k}{s_k} \right) e^{-j\xi_k t} x_i(t) dt \right|. \tag{5}$$

It denotes the correlation between the  $K_{th}$  node  $g_k(t)$  of the Gabor basal function and the  $i_{th}$  signal  $x_i(t)$ .

(2) Hidden layer: There are  $H$  nodes  $o_1, o_2, \dots, o_H$  in this layer. The output  $\phi_{ik}$  of the  $K_{th}$  Gabor basal function in the input layer is made weighted summation with the weight coefficients of neural network  $w_{kh}^{(1)}$ , as follows

$$net_{ih}^{(1)} = \sum_{k=1}^K \phi_{ik} w_{kh}^{(1)}, h = 1, 2, \dots, H. \tag{6}$$

That is the output of the  $h_{th}$  node in the hidden layer, when the input signal in the input layer is  $x_i$ .

The exciting function of the hidden layer is hyperbolic tangent function

$$f(x) = \tanh(x) = \frac{1 - e^{-x}}{1 + e^{-x}}, (-1 < f(x) < 1). \tag{7}$$

And then, the output of the  $h$ th node of the hidden layer given by the hyperbolic tangent function excited by the input  $net_{ih}^{(1)}$  is

$$o_{ih} = f(net_{ih}^{(1)}) = \frac{1 - e^{-net_{ih}^{(1)}}}{1 + e^{-net_{ih}^{(1)}}}, \quad h = 1, 2, \dots, H. \quad (8)$$

(3) Output layer: There are  $M$  nodes  $y_1, \dots, y_m$  in this layer. The input of the  $m_{th}$  node  $y_m$  is

$$net_{im}^{(2)} = \sum_{h=1}^H o_{ih} w_{hm}^{(2)}, \quad m = 1, 2, \dots, M. \quad (9)$$

And the output of the  $m_{th}$  node of the hidden layer given by the hyperbolic tangent function excited by this input is

$$y_{im} = f(net_{im}^{(2)}) = \frac{1 - e^{-net_{im}^{(2)}}}{1 + e^{-net_{im}^{(2)}}}. \quad (10)$$

### 2.3 Training Algorithm of Gabor Wavelet Neural Network

Here the improved BP algorithm is adopted. In order to get the minimal sum of square errors between expectation output vector  $T$  and the practical output of the input target vector  $x_i$ , the weight coefficients  $w_{kh}^{(1)}$  of the hidden layer, the weight coefficients  $w_{hm}^{(2)}$  of the output layer and the parameters  $s_k$ 、 $u_k$ 、 $\xi_k$  of Gabor basal function all need to be adjusted adaptably in the study stage, namely

$$E = \frac{1}{2} \sum_{i=1}^N \sum_{m=1}^M (T_{im} - y_{im})^2 \quad (11)$$

Makes  $E$  minimize.

With the relevant exciting function as hyperbolic tangent function, here are the derivatives of  $s_k$ 、 $u_k$ 、 $\xi_k$  the cost function  $E$  relative to the  $K_{th}$  Gabor basal function node

$$\frac{\partial E}{\partial u_k} = - \sum_{i=1}^K \sum_{h=1}^H \delta_{ih}^{(1)} w_{kh}^{(1)} \frac{\partial \phi_{ik}}{\partial u_{ik}} \quad (12)$$

$$\frac{\partial E}{\partial s_k} = - \sum_{i=1}^K \sum_{h=1}^H \delta_{ih}^{(1)} w_{kh}^{(1)} \frac{\partial \phi_{ik}}{\partial s_{ik}} \quad (13)$$

$$\frac{\partial E}{\partial \xi_k} = - \sum_{i=1}^K \sum_{h=1}^H \delta_{ih}^{(1)} w_{kh}^{(1)} \frac{\partial \phi_{ik}}{\partial \xi_{ik}} \quad (14)$$

where

$$\delta_{ih}^{(1)} = \sum_{m=1}^M \delta_{im}^{(2)} w_{hm}^{(2)} o_{ih} (1 - o_{ih}) \quad (15)$$

and

$$\delta_{im}^{(2)} = (T_{im} - y_{im}) y_{im} (1 - y_{im}) \quad (16)$$

Let the  $N_1$  groups observation vectors of the 1st kind of target signals are  $x_1, x_2, \dots, x_{N_1}$ , the  $N_2$  groups observation vectors of the 2nd kind of target signals are  $x_{N_1+1}, \dots, x_{N_1+N_2}$ , and the  $N_p$  groups observation vectors of the  $p$ th kind of target signals are  $x_{N_1+\dots+N_{p-1}+1}, \dots, x_N$ . In every iterative step of the training stage, weight coefficients and the parameters of Gabor basal function should be adjusted adaptably. When the network converges, the nodes of Gabor basal function denote the optimal part of time-frequency-scale 3-D space, namely for all the given training samples, the classification and decision-making of these signals by the parameters of Gabor basal function are most reliable.

### 3 Implementation of Gabor Wavelet Neural Network Based on Parallel Structure

#### 3.1 Multiple CPU Parallel Structure

According to the parallel property of neural network, we design a multi-CPU rapid target recognition system based on parallel structure. The system's hardware configuration is shown as Fig. 2, where, DSP1 is to read the image data transferred from camera, and simultaneously transfer the  $32 \times 32$  pixel lattice contained target to DSP2, DSP3, DSP4, DSP5 by HPI interface; DSP2, DSP3, DSP4 are to generate invariant code related to different targets according to some neural network algorithm; DSP5 is to calculate motion characteristics of the target (namely the magnitude and direction of the velocity); DSP6 is to make weighted summation of posteriori probability deferent from improved BP network, and display the classification result on the monitor. Based on such a multi-CPU parallel system, we have designed every tache of Gabor wavelet neural network according to different structure.

Every DSP of the hardware system is the TI digital signal processor TMS320VC5409 of high performance/price ratio. The device is a 16-bit fixed-point high-speed chip with an instruction execution time up to 100MIPS, 3.3-V/1.8-V power supply and 32K bytes internal RAM, whose peripheral hardware interface is shown as Fig. 2.



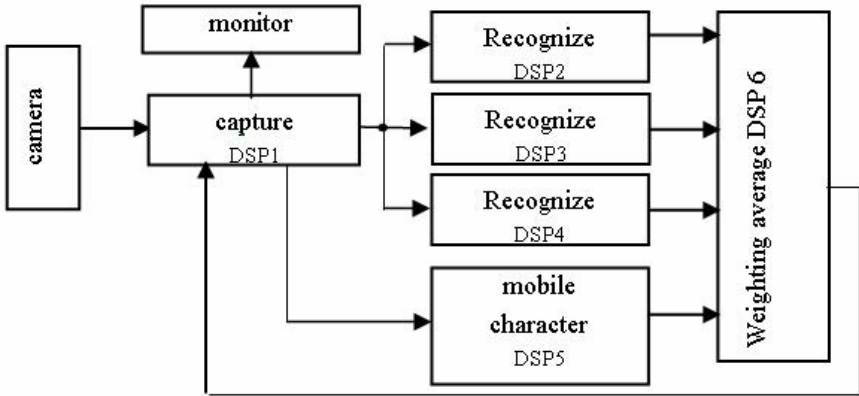


Fig. 2. Multi-CPU configuration with parallel structure

### 3.2 Algorithm Implementation and Simulated Result

The function of DSP1 is mainly about image preprocesses including image binaryzation, Gabor transform etc., which is the input layer. The main part of the algorithm includes the scaling, offset and frequency modulation parameter of Gabor transform, improved BP neural network and algorithm optimization implemented separately in parallel CPUs, which is the hidden layer. The classification result is implemented in the DSP6, which is the output layer.

Three networks of different scale are trained in the experiment; the network structures of DSP2, DSP3 and DSP4 are separately 1024-15-4, 1024-9-4 and 1024-15-4. The network of DSP4 is to distinguish plane and other flyers (birds for example), taking velocity etc. as characteristics. The network structures are set randomly, and only the former three DSPs are used here.

We choose four types of plane models as classification targets, and sample by every  $1.5^\circ$  with a total of 240 samples, even distributed in the range  $0 \sim 360^\circ$ . The part binarization samples of 4 types of plane are shown as in Fig. 3. During the experiment,

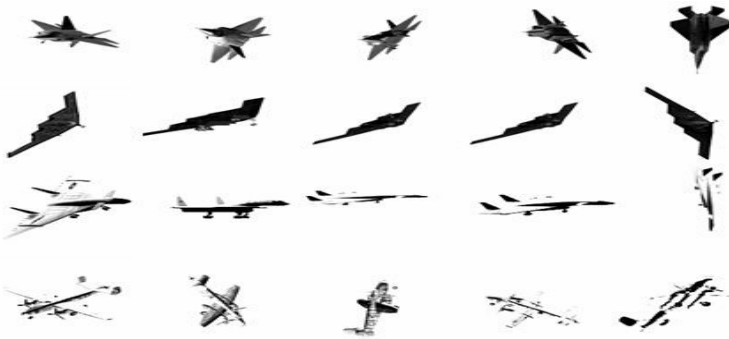


Fig. 3. Part samples of four types of plane

we select the training samples  $64 \times 4 = 240$  and the testing samples  $40 \times 4 = 160$  randomly from the sample data-base with a ratio of 3:2.

The simulation result indicates that recognizing time is less than 40ms. When the system is applied for the new targets real-time sampled in the simulating environment, the recognition ratios of four types of targets all reach 98%+. And compared with single BP network, its recognition ratio is much higher, and its generalization ability is much stronger. Suppose that the single BP network adopts the structure such as DSP2 and single network has similar performance, the recognition ratio for the trained samples is 93%. Compare results of target recognition ratio are given in Table 1.

**Table 1.** Compare result of target recognition ratio

	Training sample					Testing sample				
Hidden nodes	4	5	7	9	15	4	5	7	9	15
recognition ratio in this paper (%)	98.03	98.16	98.34	98.56	99.45	97.16	97.85	98.04	98.17	98.84
recognition ratio in BP neural network (%)	93.75	94.53	93.75	93.75	89.06	86.34	89.45	90.03	91.23	90.46

## 4 Conclusion

A novel algorithm of Gabor wavelet neural network based on parallel structure is presented in this paper, and the system of multi-target recognition is designed. Mainly based on the characteristics of multi-CPU parallel structure and the parallel nature of neural network, the algorithms of Gabor wavelet neural network is proved theoretically. The relevant algorithm structure is designed; the training and recognizing algorithms for image target recognition are constructed. According to such a parallel structure design, the algorithm of Gabor wavelet neural network exerts its parallel property veritably, and shortens the training time and recognizing time. Finally, the simulation experiment of target recognition for 4 types of plane gray images was demonstrated; the result with a 98%+ recognition rate and a 40ms recognizing time, indicated that the algorithm presented in this paper worked well, and could meet the require of engineering application.

## References

1. Zhang Q, Benveniste A. Wavelet networks. [J] IEEE Trans. on NN, 1992, 3(11): 889–898.
2. Pati YC, Krishnaprasad PS. Analysis and synthesis of feedforward neural network using discrete affine wavelet., Baskshi [J]. IEEE Trans. On NN, 1993, 4(1): 73–75.

3. Baskshi BR, Stephanopoulos G. Wave-net: a multiresolution, hierarchical neural network with localized learning. [J] American Institute Chemical Engineer Journal, 1993, 39(1): 57—81.
4. Zhang Jun, Walter G, Miao Y, et al. Wavelet neural networks for function learning. [J] IEEE Trans. on SP, 1995, 43(6): 1485—1497
5. Shi Y, Zhang X.-D. Gabor atom networks for signal classification with application in radar target recognition [J]. IEEE Trans. Signal Processing, 2001, 49: 2994—3004.
6. Kreinovich V, Sirisaengtaksin O and Cabren S. Wavelet neural networks are asymptotically optimal approximators for functions of one variable. [C] Florida, USA: Proceeding of IEEE ICNN'94, 1994. 1: 299—304.
7. Tingfa Xu etc. Multi-target recognition with improved BP algorithm. Optics and precision engineering, 2003, 10: 513—515
8. Tingfa Xu etc. Robust gray image target recognition base on multi-channel Gabor wavelet filters. Optical technique, 2004, 2: 201—203

# A Momentum-Based Approach to Learning Nash Equilibria

Huaxiang Zhang<sup>1</sup> and Peide Liu<sup>2</sup>

<sup>1</sup> Dept. of Computer Science, Shandong Normal University  
Jinan 250014, Shandong, China  
Huaxzhang@hotmail.com

<sup>2</sup> Dept. of Computer Science, Shandong Economics University  
Jinan 250014, Shandong, China

**Abstract.** Learning a Nash equilibrium of a game is challengeable, and the issue of learning all the Nash equilibria seems intractable. This paper investigates the effectiveness of a momentum-based approach to learning the Nash equilibria of a game. Experimental results show the proposed algorithm can learn a Nash equilibrium in each learning iteration for a normal form strategic game. By employing a deflection technique, it can learn almost all the existing Nash equilibria of a game.

## 1 Introduction

Game theory studies the strategic interactions among large populations of players that make decisions by considering not only their own actions but also the actions of others. In a game, a player needs to determine a course of actions for interacting strategically with others. Numerous paradigms have been proposed in the literature. Learning a Nash equilibrium (NE) constitutes a central concept in game theory. After a NE has been reached, no player can do better by changing his strategies under the condition that the other players adhere to follow the equilibrium strategy.

Several approaches, such as fictitious play [1], opponent modelling [2] and learning automata [3], have been proposed to learn the Nash equilibria for strategic games. Other learning algorithms have been proposed for infinite stochastic games, such as minimax-Q [4] and Nash-Q [5]. Most of the above mentioned algorithms guaranteed to converge to a equilibrium in the limit.

Policy gradient learning approaches have been proposed [6-8], which are mainly concerning with 2x2 matrix games. Another particular interesting issue is to learn more than one or all the Nash equilibria of a game, the above algorithms do not work for this case.

Learning the Nash equilibria of a finite strategic game remains a challenging task up-to-date, and novel approaches should be proposed. From a different point of view, NE detecting is formulated as a global optimization problem in [9] and makes some programming approaches possible to be used in finding a NE.

In this paper, we propose a novel learning technique named momentum-based approach (M-BA) and organize the paper as follows: Section 2 describes the basic concepts and the formulation of the problem. Section 3 extends the ideas

of gradient policy learning to more general form strategic games and demonstrates the learning algorithm. Section 4 represents the experimental results and concludes the paper.

## 2 Problem Formulation

A matrix game or strategic game is a tuple  $(n, A_{1\dots n}, R_{1\dots n})$ , where  $n$  is the number of players in the game,  $A_i (i \in (1, \dots, n))$  is the set of actions available to player  $i$ . The joint action set of all the players (we also call it the set of the action profile)  $A = A_1 \times A_2 \dots \times A_n$  is the Cartesian Products of all sets  $A_i$ .  $R_i$  is player  $i$ 's payoff function. Each player selects an action from his action set and receives a payoff that is a function of an action profile. We just focus on the policy learning of strategic games, as it is a fundamental case.

Based on the self-rationality assumption of game theory, each player finds strategies to maximize his own payoff, which is a function of the players' strategy profile. Let  $a_{ij}$  is an element of  $A_i$ , and  $m_i = |A_i|$  (the number of elements in  $A_i$ ). Let  $R_i$  is the set of real valued functions on  $A_i$ . We denote  $r_{ij} = r_i(a_{ij})$ , where  $r_i \in R_i$ .

Now, let  $\alpha_i$  be the probability distribution over the elements of  $A_i$ , and each term  $\alpha_{ij}$  in  $\alpha_i$  is related to an action  $a_{ij} (a_{ij} \in A_i)$ , then the following formula holds

$$\sum_{j=1}^{m_i} \alpha_{ij} = 1, \alpha_{ij} \geq 0, \forall i \in \{1, \dots, n\}, \forall j \in \{1, \dots, m_i\} \tag{1}$$

We define  $\alpha = \times_1^n \alpha_i$ , then  $\alpha \in [0, 1]^m$ , where  $m = \sum_{i=1}^n m_i$ . Each  $\alpha$  is noted as  $\alpha = (\alpha_1, \dots, \alpha_n) = (\alpha_i, \alpha_{-i})$ , where  $\alpha_i = (\alpha_{i1}, \dots, \alpha_{im_i})$  and  $\alpha_{-i}$  is the strategy profile of all other players except  $i$ .  $\bar{\alpha}_{ij}$  denotes  $(\alpha_{i1}, \dots, \alpha_{ij-1}, \alpha_{ij+1}, \dots, \alpha_{im_i})$ , and we call it the complement of  $\alpha_{ij}$ .

For any player, if he selects an action deterministically (means he selects an action with probability one), we call he implements a pure strategy, otherwise, he implements a mixed strategy. For any strategy profile of all the players, the payoff function of a player is calculated as:

$$f_i = \sum_{a \in A} \alpha(a) r_i(a) \tag{2}$$

where

$$\alpha(a) = \prod_{i=1}^n \alpha_i(a_i) \tag{3}$$

If player  $i$  plays a pure strategy  $a_{ij}$ , then the strategy profile of the game is denoted  $(a_{ij}, \alpha_i)$ . A Nash equilibrium of a game is a strategy profile of all players, and at the NE point, no player has an intention to deviate from it.

## 3 Policy Learning Approach

We extend the main ideas of M-IGA [8] to more general strategic games and make some modifications.

### 3.1 Learning in $n$ -Player Strategic Games

A  $n$ -player strategic game is formulated as a tuple described in section 2, and the expected payoff of a player can be calculated by (2) given the mixed strategy profile of the game. To describe (2) in details, we substitute  $\alpha(a)$  in (2) with (3), and (2) can be rewritten as follows

$$f_i = \sum_{a \in A} \left( \prod_i^n \alpha_i(a_i) \right) r_i(a) \tag{4}$$

We applied the gradient ascent approach to this difficult problem, and give the policy-updating rule in the following form

$$\alpha_{ij,t+1} = \alpha_{ij,t} + \eta \frac{\partial f_i(\alpha_t)}{\partial \alpha_{ij}} + \delta_{ij}(t) \gamma \Delta \alpha_{ij,t} \quad \forall i, j \tag{5}$$

$\eta$  and  $\gamma$  are two parameters,  $\delta_{ij}(t) \in \{0, 1\}$ , and is given by

$$\delta_{ij}(t) = \begin{cases} 1 & \frac{\partial^2 f_{ij}(\alpha_t)}{\partial^2 t} |_{\alpha_{ij}, \alpha_{-i}} > 0 \\ 0 & otherwise \end{cases} \tag{6}$$

$\alpha_{ij}$  should satisfy the conditions given by (1). So after all  $\alpha_{ij}$ s are updated in the  $t + 1$  step, they should be normalized

$$for \quad \forall i, j, \quad \alpha_{ij}(t+1) = \begin{cases} \frac{\alpha_{ij,t+1}}{\sum_{j=1}^{n_i} \alpha_{ij,t+1}} & \alpha_{ij,t+1} \geq 0 \\ 0 & \alpha_{ij,t+1} < 0 \end{cases} \tag{7}$$

Normalization in (7) should be implemented after all those less than zero are set to zero.

### 3.2 Learning More Than One NE

We propose an approach to learning more Nash equilibria using a deflection [9] technique, which allows multiple minimizers to be obtained in a single run of an optimization algorithm. Suppose an optimization objective function has  $l$  minimizers, and each is noted as  $x_i^*$  ( $i \in \{1, \dots, l\}$ ), the goal is to find a proper transformation function  $F(x)$ , such that the reformulated function will not obtain a minimum at  $x_i^*$  ( $i \in \{1, \dots, l\}$ ), and keeps all other minima of function  $f(x)$  locally unchanged. It was shown that the following function has the deflection property

$$T(x, x_i^*, \lambda_i) = \tanh(\lambda_i \|x - x_i^*\|) \quad i \in \{1, \dots, l\} \tag{8}$$

Inspired by the deflection property of the transformation function, we transform the expected payoff functions of the players in the following way after  $l$  Nash equilibrium  $\alpha_i^*$  ( $i = 1, \dots, l$ ) has been learned

$$F_i^l = T(\alpha, \alpha_1^*, \lambda_1) \cdots T(\alpha, \alpha_l^*, \lambda_l) f_i(\alpha) \tag{9}$$

We make a simple assumption  $\min_{a_{ij}} \min_{a_{-i}} r_i(a) > 0 (\forall i, j)$ . It implies the payoff of player  $i$  in the worst case is greater than zero. If this condition is not satisfied for a strategic game, it is easy to add a positive number to all the matrix entries, and make them greater than zero. This transformation keeps the Nash equilibria of the game unchanged. Based on the above formulation, we describe the algorithm of policy learning with momentum terms (M-BA)

**Algorithm M-BA**

- (1) Generate  $m$  random positive numbers  $b_{11}, \dots, b_{1m_1}, \dots, b_{n1}, \dots, b_{nm_n}$ , calculate the starting strategy profile  $\alpha_{ij}$  in the following way  $\alpha_{ij} = \frac{b_{ij}}{\sum_i^m b_{ij}}$  for  $\forall i, j$
- (2) loop until the number of restarts allowed is exceeded
  - do until the given criteria is reached
    - {each  $\alpha_{ij}$  is updated based on the rule of formula (5-7) }
    - once a NE is detected, store it , reformulate the expected functions of players according to formula (9) , and implement step (1)
- (3) print the detected Nash equilibria

**4 Experimental Results and Conclusions**

We evaluate the proposed algorithm M-BA by applying it to a set of benchmark game problems that are included in the GAMBIT software suit (ver. 0.2005.12.12) [10], and implement the algorithm on each test problem 50 times, and averaged the final results. The mean number of Nash equilibrium found and the mean iteration times are reported. The standard deviation is also listed.

Game 1. A three-player normal form game with two strategies for each player. This game has nine Nash equilibria, which is the maximal number of regular Nash equilibria possible for a game of this size (2x2x2.nfg)

Game 2. A three-player normal form game with two strategies per player. This game has three pure strategy equilibria, two equilibria which are incompletely mixed, and a continuum of completely mixed equilibrium. (2x2x2-nau.nfg)

Game 3. A three-player normal form game with two pure strategies available to each player. Nine equilibria are available in this game. The payoffs are given in table 1

Game 4. A two-player game with 4 pure strategies to each player. 15 equilibria are available in this game. Payoffs are given in table 2.

The number of restarts in algorithm M-BA is set to 15 for game one, 10 for game two, 15 for game three and 20 for game four. The setup of other parameters is given as

- $\lambda_i = 1 (\forall i)$
- $\gamma = 0.5$
- $\eta = \frac{1}{(1000+t)^{2/3}}$ ,  $t$  is the number of step iteration
- stop precision is set to 0.01, or the iteration number is set to 5000

**Table 1.** Payoff matrices for game 3

$s_{31}$	$s_{21}$	$s_{22}$	$s_{32}$	$s_{21}$	$s_{22}$
$s_{11}$	9,8,12	0,0,0	$s_{11}$	0,0,0	3,4,6
$s_{12}$	0,0,0	9,8,2	$s_{12}$	3,4,4	0,0,0

**Table 2.** Payoff matrices for game 4

	$s_{21}$	$s_{22}$	$s_{23}$	$s_{24}$
$s_{11}$	3,2	0,0	0,0	0,0
$s_{12}$	0,0	2,2	0,0	0,0
$s_{13}$	0,0	0,0	1,4	0,0
$s_{14}$	0,0	0,0	0,0	4,7

**Table 3.** Results for the four games

Game	Nash equilibria		Function evaluation per equilibrium					
	$\mu$	$\sigma$	min max		$\mu'$	$\sigma'$	min	max
1	8.98	0.98	8	9	2543	480.21	2012	3149
2	6	0	6	6	2364	378.95	1961	2854
3	9	0	9	9	2109	348.27	1837	2410
4	15	0	15	15	2678	439.56	1967	3011

Results in table 3 are averaged over 50 experiments for each game.  $\mu$  and  $\sigma$  represent the mean number and deviation number of different Nash equilibria learned, and  $\mu'$  and  $\sigma'$  represent the mean value and deviation value of learning iterations required to learn a Nash equilibrium. Min and max in the Nash equilibria represent the minimum(min) and maximum(max) number of Nash equilibria learned, and the minimum(min) and maximum(max) value of function evaluation represent the minimum and maximum number of iterations for learning a Nash equilibrium.

Based on the assumption of self-rationality, a player adjusts its strategy to maximize his payoff. As the payoff is a function of the mixed strategy profile of all players, updating the strategy profile makes the values of the payoff functions decrease or increase from step to step. After a Nash equilibrium has been reached, the change of the values of payoff functions will not occur, and the learning iteration stops at that time.

After a new Nash equilibrium has been learned, the payoff functions of all players have been reformulated. So a Nash equilibrium will not be learned at the previous point, and the unlearned potential Nash equilibrium points keep unchanged.

The results in table 3 show M-BA can learn almost all the Nash equilibria in all the test game problems. The number of iteration steps of learning a Nash equilibrium is less than 4000 iteration numbers. This is caused by the momentum



terms, which is used to accelerate the learning. Other intelligent algorithms [11-12] has been discussed in [13] which show they cannot compute all the Nash equilibria in complex game problems.

## References

1. O. J. Vrieze, S. H. Tij. Fictitious play applied to sequence of games and discounted stochastic games. *International Journal of Game Theory* 11, 2(1982) 71-85
2. C. Claus, C. Boutilier. The dynamics of reinforcement learning in cooperative multiagent systems, in: proceedings of 15th National Conference on Artificial Intelligence (1998)
3. K. Verbeeck, A. Nowe, t. Lenaerts, J. Parent. Learning to reach the pareto optimal Nash Equilibrium as a Team. *Lecture Notes in Artificial Intelligence* 2557
4. M. L. Littman. Markov games as a framework for multiagent reinforcement learning. In Proc: 11th international conference on machine learning. New Brunswick, NJ, Morgan Kaufmann, San Mateo, CA (1994) 157-163
5. J. Hu, M. P. Wellman. Nash Q-Learning for General-Sum Stochastic Games. *Journal of Machine Learning research* (2003) 1:1-30
6. M. Bowling, M. Veloso. Multiagent learning using a variable learning rate. *Artificial Intelligence* 136 (2002) 215-250
7. S. Singh, M. Kearns, Y. Mansouv. Nash convergence of gradient dynamics in general-sum games. In proceedings of the 17th conference on uncertainty in artificial intelligence (2000) 541-548
8. H. Zhang, S. Huang. Convergent Gradient Ascent with Momentum in General-Sum Games. *Neurocomputing* 61(2004) 449-45
9. R.D. Mckevey, A Liapunov function for Nash equilibria. Technical Report, California institute of Technical Report, California Institute of Technology (1991)
10. R. D. Mckelvey, A. McLennan, T. Turocy. *Gambit Command language*, California Institute of Technology (2000)
11. N. Hansan, A. Stermer. Completely derandomized self-adaption in evolution strategies. *Evolutionary Computation* 9, 2 (2001) 159-195
12. J. Kennedy, R. C. Eberhart, *Swarm intelligence*, Morgan Kaufmann Publishers, Los Altos, CA (2001)
13. N. G. Pavlids, K. E. Parsopoulos, M. N. Vrahatis. Computing Nash equilibria through computational intelligence methods 175 (2005) 113-136

# Model of Emotional Agent

Jun Hu<sup>1</sup>, Chun Guan<sup>1</sup>, Maoguang Wang<sup>2</sup>, and Fen Lin<sup>2</sup>

<sup>1</sup> Department of Computer Science and Technology, Nanchang University,  
Nanchang 330029, China  
hujun@ncu.edu.cn

<sup>2</sup> Institute of Computing Technology, Chinese Academy of Sciences,  
Beijing 100080, China

**Abstract.** The research result of neurobiology shows that emotions have a great effect on the intelligence of human beings. To make agent have the ability of handling emotions, a new model structure of emotion agent is proposed, based on the traditional BDI agent model. The model firstly establishes a new emotion knowledge base for emotion expression, secondly performs emotion reasoning by an emotional reasoning algorithm, and finally implements the emotions treatment. An emotional agent based on this model has been realized, experiment results show that it is efficient to transact simple emotions.

**Keywords:** Emotional agent, model, knowledge base, emotional reasoning.

## 1 Introduction

Emotions of human are very complex and widely spread. The research result of neurobiology shows that emotions have a great effect on the intelligence of human beings [1]. Therefore, the research on emotional agent helps to improve the characteristic of intelligent of agent [2]. The explanation for emotion in psychology is that emotions are evaluations for oneself or for relation status between agents and environment by human body. The emotional reaction of human beings is generated by stimulation of outside environment and affected by the mechanism of demand and requirement inside themselves. On the view of agent, emotions are evaluations and reactions for current interior status and the relation status between agent's desire and plan by agent in exterior environment. Hence, based on the traditional BDI agent model, a new model structure of emotion agent is proposed to make agent have the ability of handling emotions. The model firstly establishes a new emotion knowledge base for emotion expression, secondly performs emotion reasoning by an emotional reasoning algorithm, and finally implements the emotion treatment.

## 2 Emotional Agent Model

A new emotional agent model is proposed, which is composed of belief, desire, intention, plan, rational reasoning machine, emotional knowledge base and emotional

reasoning machine. In order to treat emotion, it extends the traditional BDI agent model by adding an emotional knowledge base and an emotional reasoning machine. The structure of model is illustrated in Figure 1.

In this model, belief, desire, intention and emotion are represented by symbols. The rational belief of emotional agent is the basic and most important element. Some other problems such as the representation and rational deduction of mental status all base on and rely on the rational belief. The belief of emotional agent usually can be regarded as the rational knowledge base. It contains abundant contents including the basic axiom, the domain axiom familiar by emotional agent and the understanding of facts and data in the domain, which is constructed by the facts of the outer world, the exterior environment and interior status. These facts are described by one order logic.

Desire is the initial motivation of agent. It is the state for agent to reach or state set to keep, which is the goal set to implement. The plan and action are activated by desire, which can be described as a kind of exception and judgment. The desire's implementation is judged by the state's coming into existence.

The intention is the most needed or suitable one to accomplish among the desire promising to implement currently. It is the prearrangement for agent's action. Current intention directs current action. Intention differs from goal. It is the goal or sub goal to execute. The goal is stable relatively while intention is easy to change. The action intention of emotional agent is their action queue. Its execution is associated with the schedule of emotional agent. Once the action is scheduled, the intention switches to the execution of action.

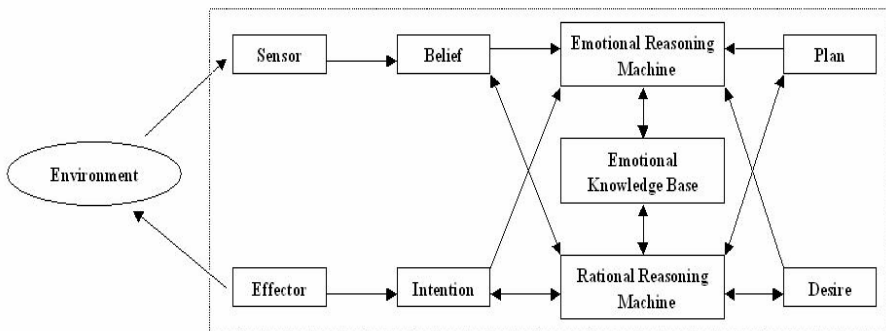


Fig. 1. A model of emotional agent

Plan is the action program of emotional agent. It points out the main method for the emotional agent to realize certain requirement. Current agent itself and environment state can be connected with agent's goal. It becomes the main means leading agent to implementing the requirement. In fact, plan associates the emotional agent's rational belief and the capability of behavior description with goal organically. It points out which action will adopt to realize certain goal under certain rational belief.

Sensor is responsible for receiving the message of environment and changing the belief of agent. It performs the action and sends message to outside environment according to the belief of agent.

The function of rational reasoning machine is consistent with that of traditional BDI model. It implements intention reasoning according to belief, desire and plan, implements rational logical reasoning irrelevant with emotional reasoning such as the logic reasoning for one order predicate.

Emotional knowledge base stores the facts relevant with emotion, which are expressed with one order logic. In the model, the facts irrelevant with emotion are expressed with rational belief and the facts relevant with emotion are described in emotional knowledge base.

Emotional reasoning machine is used to dealing with the logic reasoning relevant with emotion. The emotional reasoning is implemented according to surrounding environment, plan, desire and situation of current intention. In the model, emotional reasoning is built on the rational reasoning but there is difference between them. In the structure of emotional agent, rational reasoning machine determines current intention and action according to belief, plan and goal, while emotional reasoning machine implements emotion reasoning utilizing emotional knowledge base according to the status of agent's desire, plan, belief and current intention.

In the model of emotional agent, agent has initial desire and plan for goal to arrive. When the surrounding environment has changed and after it influences agent's belief, rational reasoning machine implements some intention by logical reasoning. Emotional reasoning machine implements emotional reasoning according to belief, desire, plan and realized intention. At the same time, emotional knowledge also can be the premise of rational reasoning. We build the emotional agent model with ration and emotion coexistent, dependent and different, which can effectively implement the emotional expression and reasoning.

### 3 Emotional Knowledge Base

An emotional knowledge base of agent is a three-tuple  $\langle T, S, B \rangle$ ,  $T$  is the set of basic concepts in the emotion domain,  $S$  represents the cause - effect relationship among facts in the emotion domain by formula,  $B$  denotes current facts in the emotion domain. But how can we describe emotion facts and concepts? A new method is presented.

Assuming that there is  $N$  emotion types, we use a vector space  $E$  to represent the emotions,  $E = \langle e_i \mid i=1, 2, \dots, N \rangle$ , where  $N$  represents the number of the emotion types. For the fuzzy intensity of emotion we use a discrete variable  $V_i$  and divide it into  $M$  levels.

Then emotion model can be represented simply as:

$$\begin{pmatrix} E \\ V \end{pmatrix} = \begin{pmatrix} e_1 & e_2 & \dots & e_n \\ v_1 & v_2 & \dots & v_n \end{pmatrix}$$

Once the types of emotion is fixed, the emotion model can be simplified as a vector  $E = \langle v_1, v_2, \dots, v_n \rangle$ .

In the light of the psychology research we divide the emotion into five basic types: anger, afraid, happy, sad, dislike. Thus the agent emotion can be represented as  $E = \langle v_{\text{anger}}, v_{\text{afraid}}, v_{\text{happy}}, v_{\text{sad}}, v_{\text{dislike}} \rangle$ . To simplify the representation we just divide the emotion intensity  $V_i$  to five levels,  $V_i = (0, 0.25, 0.5, 0.75, 1)$ .

For instance,  $E = \langle 0, 0, 0, 0, 0 \rangle$  represents that the autonomic unit has no any emotion preference.

$E = \langle 0.25, 0.5, 0, 0, 0 \rangle$  represents the autonomic unit has the complex emotion of a little angry also very fear.

## 4 Emotional Reasoning Algorithm

The emotional reasoning machine implements emotion reasoning utilizing the emotional knowledge base and the emotional reasoning algorithm according to the status of agent's desire, plan, belief and current intention. The emotional reasoning algorithm is followed:

- Step1 According to belief, desire, plan and realized intention, searching sub-goals that has been realized and unrealized sub-goals that should be realized since  $\text{Time}(i-1)$ .
- Step2 Making reasoning by one order logic rule and obtaining corresponding emotional vector in the knowledge emotion base.
- Step3 Calculating the highest value of every emotion vector as the current emotion in  $\text{Time}(i)$ .

Now we evaluate the task result and make reasoning based on emotion representation.  $\text{task1} \rightarrow \langle 0, 0, 0.25, 0, 0 \rangle$ , which means the emotional agent is a little happy, namely the result is a little satisfactory when executing the task.  $\neg\text{task2} \rightarrow \langle 0, 0, 0, 0.75, 0 \rangle$ , which means it is very sad because it can not complete the task in time.

After executing the specified task, emotional agents will obtain the emotion vector, and we will calculate the highest value of every emotion vector, for instance if at moment of  $\text{Time}(i-1)$ , the emotional agent has achieved the goals of B and C, but failed in the goal of A. We have produced the emotion vectors as following:

$$\begin{aligned} \neg A &\rightarrow \langle 0, 0, 0.25, 0, 0 \rangle, \\ B &\rightarrow \langle 0, 0, 0.5, 0, 0 \rangle, \\ C &\rightarrow \langle 0, 0, 0.75, 0, 0 \rangle. \end{aligned}$$

Then we get the final emotion  $E = E1 * E2 * E3 = \langle 0, 0, 0.75, 0, 0 \rangle$ , where \* means the synthesis operator. Namely, it has the more satisfactory degree for the task result. Certainly we can add the weights for the specified task to represent the task priority so that the emotion representation for the satisfactory degree can be more exact.

## 5 Experiments

AGrIP (Autonomic Grid intelligent Platform) [5] is based on the FIPA-compliant multi-agent environment MAGE [6]. As a distributed development platform, the MAGE environment comprises three tools: Modeling Tool for Multi-Agent System (MTMAS) supports the domain analysis and design including AUMML modeling tool, Visual Agent Studio (VAStudio) is a multi-agent development platform, MAGE provides the deployment and running environment for the multi-agent system.

Our experiment is to make use of emotional agent and AGrIP to build an autonomic grid environment, in which emotional agents can generate some simple emotions automatically and decide more efficient cooperative policy to obtain resources according to different emotion. The autonomic grid includes some physical nodes. The calculation task is distributed to different nodes running the MAGE platform. Every emotional agent, as a basic autonomic grid node, will monitor the network and local hardware resource usage including CPU, memory, disk and network utilization. When distributing the sub-task to the nodes according to the plan, emotional agent automatically generates some simple emotions according to its resource usage, goal, plan and realized intention, then determines different cooperative policy including positive policy, negative policy and usual policy for obtaining resources efficiently according to different emotion. During the running process every emotional agent will give its visual emotion representation. Figure 2 depicts the experiment process.

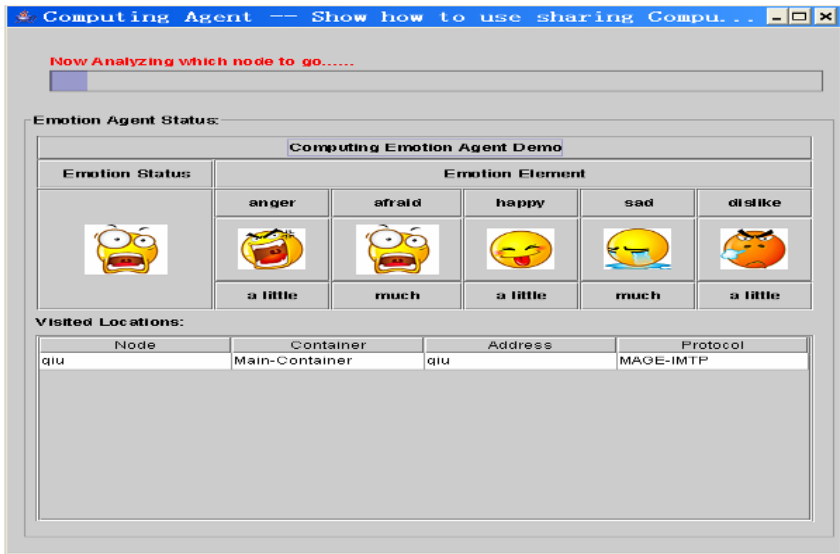


Fig. 2. Dynamic emotion response

It shows that the emotional agent qiu is negotiating with the other agent for the running resource by positive cooperative policy. Agent qiu want to migrate other node because of the deficiency of running resource. After they cooperate to finish the specified task, it will feed back the result.

## 6 Conclusions

In the past emotions have been dismissed as a distraction to the logical, scientific thought process. More recently however, the importance of emotion in human-like intelligence has been identified. The research on emotional agent helps to improve the characteristic of intelligent of agent. To make agent have the ability of handling emotions, a new model structure of emotion agent is proposed, based on the traditional BDI agent model. The model firstly establishes a new emotion knowledge base for emotion expression, secondly performs emotion reasoning by an emotional reasoning algorithm, and finally implements the emotion treatment. An emotional agent grid based on this model has been realized, experiment results show that it is efficient to transact simple emotions. We are doing further research on the complicated emotion expression and reasoning for efficient cooperation among agents in a grid environment.

**Acknowledgments.** This work is supported by the National Basic Research and Development Plan of China (Grant No. 2003CB317004) and the National Science Foundation of China (Grant No. 90604017).

## References

1. Damasio, A.: *Descartes error: emotion, reason and the human brain*. New York: Avon Books (1994)
2. Joseph, B.: The role of emotion in believable agents. *Communications of the ACM* 37(7) (1994) 122–125
3. Karthi, S., Debbie, R.: The use of emotions to create believable agents in a virtual environment. *Proceedings of the fourth international joint conference on autonomous agents and multi-agent systems* (2005) 13–20
4. Oliviera, E., Sarmento, L.: Emotional advantage for adaptability and autonomy. *Proceedings of the second international joint conference on autonomous-agents and multi-agent systems* (2003)
5. Maoguang Wang, Jiewen Luo, Fen Lin, Zhongzhi Shi.: *Internet intelligent platform-AGrIP. Artificial Intelligence Applications and Innovations II* (2005) 363–370
6. Zhongzhi Shi, Haijun Zhang, Yong Cheng, Yuncheng Jiang, Qiuqian Sheng, Zhikung Zhao.: *MAGE: an agent-oriented programming environment*. *IEEE ICCI* (2004) 250–257
7. Zhongzhi Shi.: *Intelligent Agent and Its Applications*. Press of China Science (2001)
8. Gratch, J., Marsella, S.: A domain independent framework for modeling emotion. *Journal of Cognitive Systems Research* 5 (2004) 269–306
9. Zhongzhi Shi, Youping Huang, Qing He.: *MSMiner-a developing platform for OLAP. Decision Support Systems* (2005)

# Multi Region-Tree Based Dynamic Commission Home Proxy Communication Mechanism for Mobile Agent

Zehua Zhang and Xuejie Zhang

Department of Computer Science & Engineering, Yunnan University, Kunming , 650091,  
P.R. China  
{zehuazh, xjzhang}@ynu.edu.cn

**Abstract.** In order to realize collaboration and negotiation among MAs (Mobile Agents) in MAS (Mobile Agent System), or control the Mobile Agent and release the resources it occupied, There must be a reliable and efficient communication mechanism for mobile agent, it plays a very important role in the realization of MAS and in the promotion of the widely use of MAS in the large scale network environment. In this paper, we present a novel communication mechanism MRTBDCHP (Multi Region-Tree Based Dynamic Commission Home Proxy Communication Mechanism) for Mobile Agent, it can temporarily commit a server as its communication support point near the resource or its associators, it overcomes many shortcomings of the currently existing mechanisms, further more, it is more advantageous for the application of MAS in the large scale network environment, and this mechanism is more conform with the design principles of distributed computing systems.

## 1 Introduction

MA (Mobile Agent) is a special kind of software agent, it can represent its owner or a program to perform assigned task while moving from one node to another node in the complex and heterogeneous network [1,2].Communication is an essential ability for MA; a reliable and efficient communication mechanism plays a very important role in the realization of MAS and its efficiency.

Existing MA communication mechanisms can be classified as: HP (Home Proxy) [3,4], MB (Mailbox-Based) [5,6], PF (Path Forward) [7,8], RB (Region-Based) [9,10]and BC (Broadcasting) [11].Most of these mechanisms don't consider the practical application and running environment for MA, and have problems in efficiency, fault-tolerance, and not provide a good support for MA's action.

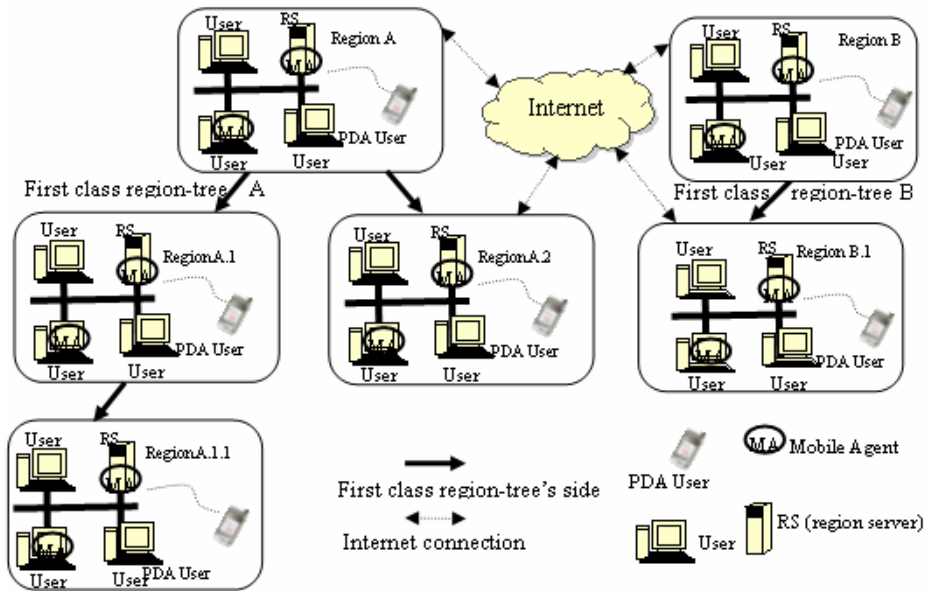
In this paper, we present a novel communication mechanism MRTBDCHP (Multi Region-Tree Based Dynamic Commission Home Proxy Communication Mechanism) for Mobile Agent, it can temporarily commit a server as its communication support point near the resource or its associators, it overcomes many shortcomings of the currently existing mechanisms, and meets the communication requirements of MA better.

The rest of this paper is organized as follows: Section 2 presents the communication mechanism MRTBDCHP. Section 3 analyze the performance of MRTBDCHP and finally conclude the paper and gives directions of future work in Section 4.



## 2 The MRTBDCHP Communication Model

**Structure of The Region-Tree:** In FIPA2000(Specification of Foundation for Intelligent Physical Agents 2000), A region is a set of agent systems that have the same authority, it consists of some nodes which link to one another, each node can run MASs in it or not. It's reasonable to divide the network into regions because of the authorizations, security mechanisms, or which department it belongs to and where it locates in.



**Fig. 1.** The structure of Multi region-tree based MA communication model

In MRTBDCHP, each region has a RS (region server) which maintains the addresses of the MASs running in this region and the addresses of MASs which birth in this region, the RS also responsible for the delivery of the messages to the MASs who reside in this region or forwarding these messages, and sends out messages for them.

The structure of the region-tree in MRTBDCHP as shown in Fig. 1, all nodes running MAS are divided into many first class regions (according to the node's location or the management requirements, etc.), each first class region is divided into many sub-regions, and each sub-region can be divided into many more smaller sub-regions continuously, every node in the tree is a RS who manages the communications for the MASs register to this region, a leaf node consists of many nodes running MAS (a node can run many MASs). A non-leaf node can include many sub-regions and many nodes. Thus a region-tree is formed, and each region in it can be a sub-region-tree.

The nodes running MASs in Internet can be organized into a system which consists of many first class region-trees logically, a first class region-tree can be a big or small one, and it needn't maintain any information of the other first region-trees.

We assume that the reasonable running environment for MA is Internet, besides the RS's logical relationship in the region-tree; the RSs are connected to each other through Internet. In the initialization phase, a server can register to another sever or accept another server's register and become a RS node in the region-tree, according to the management requirements, distance or authority, etc. Hosts in the network register to a RS according to their requirements.

**Name Scheme and Related Facilities:** Addresses of a RS or a host is an IP address. A RS responsible for the lookup of address of MAs which have committed it as the proxy (the MA in this region can choose to commit this RS is the proxy or not), and deliver the messages to these MAs directly or forwarding the messages.

In order to guarantee the correctly delivery of message, there must be a globe name space, and the name in the globe space is unique. When a MA is created, its unique name is like:

Birth node's IP	Birth node's port	PNN	MA's ID in Birth MAS
-----------------	-------------------	-----	----------------------

PNN is the programmer named name, MA's name is unique since MA's ID is unique in the MAS, MA1's name in the region A.1.1of Fig. 1 can be named is:

<167.211.127.10><3121><Buy-Camera><MA1>

It is resolved as: MA1 is birthed in MAS with Port number 3121 in the node 167.211.127.10, the programmer named MA1 "Buy-Camera" according to its task.

A RS maintains two tables, table LMA has the information of MAs who birthed in the local nodes which belong to this region (but not the MAs birthed in the sub-trees), LMA contains {MAID, Moved, Location, InMoving} entries, the field MAID is the MA's identification; when MA left the birth region and has moved to another region, Moved field is set to 1, otherwise it's 0; when the Moved field's value is 1, Location field's value is the address of the MA appointed CHP (Commission Home Proxy), otherwise it's value is 0; when the MA is in moving, InMoving field's value is set to 1, otherwise it's value is 0. Table VMA has the information of visited MAs who are not birthed in this region, VMA contains {MAID, Location, InMoving} entries, Location field's value is the address of the node (directly belong to this region) where the MA currently resides in, the other two fields have the same meaning as in VMA.

**MA's Move Mechanism:** When a MA is created on the home node, it commits the home node's RS is the Home Proxy (abbreviated as HP), and add a LMA record in the HP. When MA leave the home region, it choose to commit a RS in a first class region-tree as the CHP, this commit can according to its goal, its current location, the collaboration model or the network's circumstance, etc. This is to say that a MA can determine by itself to choose a RS as its CHP in the first region-trees according to the factors which effecting it to accomplish its task, the RS of the region where the MA directly resides in (the smallest region MA belongs to) may not be committed is the

CHP, and MAs have the same directly reside region (abbreviated as DRS) can commit different RS is their CHP. This commission is dynamical, according to the factors effect MA to accomplish its task.

IF a MA decide to leave the HP's region, the InMoving field of corresponding record in HP's LMA is set to 1, when the MA firstly arrives at a new DRS, it register on the DRS's RS by add a VMA record in the RS, then it choose a certain RS as the CHP (it can also be the DRS). After done this, a ACK message with the CHP's IP address is send back to HP, HP's corresponding record's field Moved is set to 1, InMoving is set to 0, Location field value is set to the address of the CHA. The CHP add a VMA record, Location field's value is the IP address of the DRS (it's the IP of the DRS's RS, same in the following text) where the MA currently resides in.

The move of MA is classified into three kinds: intra-DRS, intra-CHP and inter-CHA according to the scope of the move.

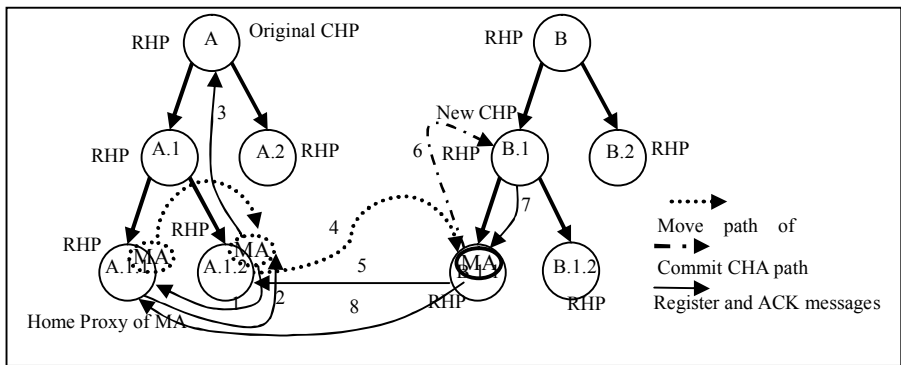


Fig. 2. The mechanism of inter-CHP move in MRTBDCHP

An example of inter-CHP move shows in Fig. 2, a MA is created in region A.1.1's node, RS A.1.1 is its HP. When MA left region A.1.1 and moved to region A.1.2, considering it may move frequently in the first class region-tree A, A is committed is the original CHP, but when MA decide to move to the new region B.1.1 in first class region-tree B, as the sequence numbers mark in Fig. 3, MA sends request message to its HP and gets the permission to move (1,2), it sends a cancel register message to previous CHP to log-out (3), MA moved to region B.1.1 (4) and sends a ACK message to A.1.2 (5), since the target resource is in region B.1 and its sub-tree, MA commit B.1 as the CHP and receive the acceptance ACK (6,7), the message with the address of the new CHP B.1 is send to MA's HP to renew the corresponding record of the MA (8).

**Message Delivery Mechanism:** If a message is send out by a MA (it can also be the messages send out by a static agent or other object) from a MAS on a node, from the PDA or other network facilities, message is firstly delivered to the MAS's communication module, then the message is send to the RS which the sender now belongs to, the RS's communication module resolve the receive MA's name and

lookup the LMA and VMA, if it is found in LMA, message is directly send to the receiver, otherwise, by consult the receiver's HP or appointed beforehand with its associators, message is delivered to the receiver's CHP or HP.

When a MA decide to Move, it sends a move request to DRS, DRS modifies the corresponding record's InMoving field's value in VMA or LMA to 1, let the time that when DRS receive the message  $T_d$  be the dividing point, messages received before  $T_d$  is send to the former address of the nodes MA now resides in, and MA must receive these messages before move; message received after  $T_d$  is cached in DRS, When MA moved to a new node successfully, it sends ACK knowledge to the former DRS and the former node it has resided in, and receive the cached messages in DRS if there were any. In case of intra-DRS move, the corresponding field InMoving's value in DRS is set to 0, and set the Location field's value to MA's current address, in case of intra-CHP and inter-CHP move, MA log-out and delete its information in the former DRS, but MA must send ACK message to its HP if it perform a inter-CHP move, and receives the cached messages in HP if there were any. There can also be occasions that the sender's communication model can not receive the ACK, then it can send the message to the receiver MA's HP, messages can also be cached in HP. Thus solve the communication failure and message chasing problem.

### 3 Performance Evaluation

MRTBDCHP is more efficient. MA needn't register to the home proxy when perform the intra-DRS and intra-CHP move except that the MA committed a new CHP, MAS can communication with each other even not look up or register to the HP.

MRTBDCHP is a model that let the communication support point (the RS or the CHP) locates in the area where the resources the MA moves to (this is the main advantage of MA), or near MA and its associators.

Dynamic adaption is another advantage of MRTBDCHP. MA can react with the change quickly, and decides by itself to commit any RS in the Region-Tree as its CHP which can have the size as small as a RS or as big as a first class region-tree. It's a dynamic, selective and layered granularity partition.

MRTBDCHP provides good fault tolerance and robust, and it has low dependence on single point. MA's register and message cache don't depend on a single point any more; it has been dispersed to the HP and DRSs, this decrease the possibility of the system breakdown caused by single point failure. Once a nod (i.e. a RS) is inaccessible, it only need to make the MAS belong to this RS and the RS's son node to choose other RSs to register in, the Region-Tree has the self-cure ability.

We have done the simulative experiment in our campus LAN by use IBM Aglets 2.0.2 and JDK 1.3.1. Compare with other existing mechanisms, experiment result manifested that with the increase of the messages, the increase of the communication cost and communication time of MRTBDCHP's is the smallest, this consistent with the analysis that the overhead and communication time caused by communication can be decreased greatly owing to the MA's dynamic and optimal commit of its CHP. The detail isn't given here because of the space limitation.

## 4 Conclusion and Future Work

A novel communication model MRTBDCHP is presented in this paper; it meets the requirements of MA communication better compare with existing mechanisms, more over, it provides a good support for MA's action, a good communication mechanism should not diminish the benefit brought by the use of MA.

MA's communication has a close relation with the collaboration and negotiation among MAs, how the effect each other will be further studied in our future work.

## Acknowledgement

This work is supported by the National Natural Science Foundation of China (NSFC) (No.60573104).

## References

- [1] D. B. Lange, M. Oshima. Seven good reasons for mobile agents. *Communications of the ACM*, 1999,42 (3),pp.88-89
- [2] G. Cabri, L. Leonardi, F. Zambonelli. *Mobile Agent Technology:Current Trends And Perspectives*. AICA' 98,1998.
- [3] D. B. Lange and M. Oshima. *Programming and deploying Java mobile agents with Aglets*, Addison-Wesley, 1998.
- [4] D. Milojicic, et al, "MASIF: the OMG mobile agent system interoperability facility", MA'98, LNCS1477, Springer, September 1998, pp.50-67.
- [5] X.Y. Feng, J.N. Cao, et al, "An Efficient Mailbox-Based Algorithm for Message Delivery in Mobile Agent Systems", LNCS 2240, Spintger-Verlag, 2001, pp.135-151.
- [6] J. Cao. Liang. Zhang and J. Yang, "A Reliable Mobile Agent Communication Protocol", ICDCS'04, pp.1063-6927/04. IEEE 2004.
- [7] J.Y. Zhou, Z.Y. Jia and D.X. Chen, "Designing Reliable communication Protocols for Mobile Agents," *Proceedings of the 23 rd internationalConference on Distributed Computing Systems Workshops (ICDCSW'03)*.
- [8] H.Hong.Tsai, F.Y..L and W.K.Chang, "Distributed SendBox Scheme for Mobile Agent Communication,"( ICMB 2005),pp.545-550.IEEE 2005.
- [9] S. Lazar, I. Weerakoon, D. Sidhu, "A Scalable Location Tracking and Message Delivery Scheme for Mobile Agents," *Seventh International Workshop on Enabling Technologies: Infrastructure for Collaborative Enterprises*, pp.243-249, IEEE 1998.
- [10] SungJin Choi, MaengSoon Baik, ChongSun Hwang, "Location Management & Message Delivery Protocol in Multi-region Mobile Agent Computing Environment", *Proceedings of the 24th International Conference on Distributed Computing Systems (ICDCS'04)*, pp. 476-483, IEEE 2004.
- [11] Murphy and G. P. Picco, "Reliable Communication for Highly Mobile Agents", *Agent Systems and Architectures/Mobile Agents (ASA/MA)'99*, October 1999, pp. 141-150.

# Research on Modeling and Description of Software Architecture of Cooperation-Oriented System

Munan Li, Hong Peng, and Jinsong Hu

School of Computer Science & Engineering,  
South China University of Technology,  
Wushan Road, 510640, Guangzhou, China  
nells\_li@hotmail.com,  
{mahpeng, cshjs}@scut.edu.cn

**Abstract.** Aimed at the cooperation-oriented software system, this paper advanced a novel model of software architecture based on cooperation. Even as the component is the aggregation entity of objects, the cooperation is defined as the aggregation of agents based on the common tasks and the environmental constraints. As to the description of cooperation-oriented architecture, this paper brought forward a solution based on the extended WRIGHT. This new language of architecture description is named as WRIGHT\*.

## 1 Introduction

The most popular models of SA (Software Architecture) are based on the philosophy of component [1] [2]. In fact, the software systems have been stepping into the times of cooperative and adaptable computing. With the development of CSCW, the cooperation-oriented system has become more and more popular, e.g. E-business based on cooperative game theory, visual enterprise, supply chain management, E-learning etc. Therefore, the cooperation-orientation is the philosophy to take the cooperation as the main design facet surrounded by other facets. In general, the model of software architecture is a three tuple: <components, connectors, configurations>, and most of the architecture descriptions are based on this model [2]. The main difference of these two concepts is that the component is apt to be the entity of computation; however, the agent is not only a computing entity, but also has such characters as mobility, intelligence and mind properties etc. Therefore, the agent is larger than the component from the view of the granularity of modeling.

In reference [3], authors ever advanced an ADL (architecture description language) for multi-agent system. They thought that the traditional ADLs based on component are difficult to describe the architecture of MAS, therefore, they brought forward an ADL based on <connector agent, computation agent>. Here, the cooperation-oriented system can be taken as one of the application frameworks of agent-oriented technology. In the same meaning, the component-oriented system is the practical implementation of OO (object-oriented) theory. Although the OO technology embodies the conceptions on cooperation and synchronization, the granularity of class or component is too small to describe the complex applications based on self-organization and cooperative computing and reasoning. One obvious difference between agent and component may lie in the philosophy of modeling on the real

application. The component is the approach or the fruit of code reuse, and is to promote the efficiency and scalable of software system. But agent is used to encapsulate the tasks and responsibilities of system. Although single agent has limited capacity of computation, the collective of agents can perform much more complex functions or tasks through negotiation, coordination and cooperation etc. In general, the components are designed to encapsulate the common business logic, i.e. mutual functions of application system. For example, the EJB and COM+ are deployed on the application server to be invoked by the different clients. The agent has much larger granularity than the general component, e.g. the spiders of Internet search, the robots of Martian explore etc. The definition of agent has “weak notion” and “strong notion” [4]. Here, we propose a simplified definition: agent is an entity with the intention and capabilities of cooperation, with the limited abilities of computing and reasoning, and with the responsibilities of role.

**Definition 1.** Agent is a three-tuple:

$$\begin{cases} Agent := \langle R, \Omega, \Phi \rangle \\ R := \{ \langle role \rangle \langle responsibility \rangle \}^* \\ \Omega := \{ \langle Computation \rangle \langle Reasoning \rangle \langle Cooperation \rangle \}^* \\ \Phi := \{ \langle Belief \rangle \langle Desire \rangle \langle Intention \rangle \}^* \end{cases} \quad (1)$$

In definition 1, R is the set of responsibility  $\Omega$  is the set of capability including cooperation and reasoning,  $\Phi$  is the set of mind properties, e.g. BDI (beliefs, desires and intentions).

As to the definition of cooperation, in the reference [5], authors ever thought that the cooperation is to assign tasks, resource and information among teamwork agents to improve the survival capacity, performance of system and to eliminate the conflicts. We are apt to adopt the concept: the cooperation is a natural phenomenon to complete some tasks which cannot be completed by any individual in a collective of agents.

**Definition 2.** Cooperation is a three-tuple:

$$\begin{cases} Cooperatin := \langle T, A, B, E \rangle \\ T := \{ \langle task \rangle \langle goal \rangle \}^* \\ A := \{ \langle Agent \rangle \}^* \\ B := \{ \langle behavior \rangle \}^* \\ E := \{ \langle constraint \rangle \langle situation \rangle \}^* \quad constraint := \{ \langle rule \rangle \langle specification \rangle \}^* \end{cases} \quad (2)$$

In definition 2, T is the set of tasks and goals, A is the set of agent, B is the set of behavior, E is the constraints of environment. Wooldridge and Jennings [6] ever described the process of cooperation in multi-agent system based on modal logic. They thought that the process of cooperation includes three steps: 1) to recognize the potential of cooperation, 2) to form the team of agent, 3) to plan the collective behaviors and 4) to do the tasks.

From the definition 2, T can be thought as the preconditions of cooperation in that the common tasks and goals are made for the potential of teamwork. After the initialization of teamwork, the team of agent can be organized, i.e. the set of A can be formed. Through the partial or global planning of behavior, the queue of actions can

be shaped. When the actions are implemented by the team, the influence on environment can be recorded. On the other hand, the cooperation among agents needs the third-part support, for example, it needs the support of immigration of agent, the intercession of conflict, the balance of resource competition, the maintenance of shared knowledge and data. Therefore, the cooperation needs to consider the mechanism of cooperation support. Here, we bring forward a formal definition of cooperation support.

**Definition 3.** Cooperation support is to provide such mechanisms or service as: the communication support, intercession support, coordination etc. Therefore, the cooperation support can be defined as the service provider.

$$\begin{cases} \textit{Cooperation\_support} := \langle \textit{services}, \textit{customers}, \textit{constraints} \rangle \\ \textit{service} := \{ \langle \textit{port} \times \textit{content} \rangle \}^* \\ \textit{customers} := \{ \langle \textit{Agent} \rangle \}^* \\ \textit{constraints} := \{ \langle \textit{rule} \times \textit{specification} \rangle \}^* \end{cases} \quad (3)$$

In the standard on software architecture of IEEE1471 [7], the SA is defined as the basic structure of software system, which uncovers the mutual relationships among ingredients of system, the relations to environment, and which directs the design and evolution of system. Here, we provide a model of software architecture on cooperation-oriented system in definition 4.

**Definition 4.** The model of software architecture of cooperation-oriented system is based on cooperation.

$$\textit{SA\_CO} = \{ \textit{cooperation}, \textit{supporters}, \textit{configurations} \} \quad (4)$$

In definition 4, cooperation is the basic element of modeling, supporter is the unit of cooperation support, and configuration is the constraints and rules. In essence, SA\_CO is based on the agent-oriented theory and method, and the cooperation is the encapsulation of agents by means of responsibility aggregation. This generalized encapsulation has advantages on those cooperative applications system. For example, the virtual manufacture, war-commanding simulation and other distribute collaboration systems. From the view of software engineering, the philosophy of cooperation can simplify the process of system design, and the analysis of domain can be isolated from the implementation, therefore, the domain specialists can complete their design without any computer background. The system designers just need to integrate the units of cooperation into a real application system.

## 2 Architecture Description and Architecture Description Language

The description methods of architecture can be classified into such two categories: informal and formal. The informal methods depend on the diagrams. Compared to the formal description based on formal language, the method of diagram is not complete and detailed to describe the software architecture [18], and is difficult to transfer the whole philosophy of design among the project members. In the light of the formal



logic fruits, the ADL is a good tool or platform to intercourse and standardize the thoughts of architecture design. Perhaps, some scholars wish to find a perfect ADL which is easy to be understood, flexible and adaptable; but so far, it is not practical. There are still arguments on responsibilities of ADL and its related tools, whether the ADLs should cover the whole process of software engineering, even directly produce the real system of application via the mechanism of language compiler? We thought it is not practical to replace the real activities of software development using a single formal method and its related tools. In fact, the software architecture is the highest level of media of design knowledge and experience. Therefore, the essence of software architecture is the reuse of the knowledge of system analysis and design, and the ADL is an efficient tool for interaction in open situations of software development. In the standard of IEEE 1471, the ADL is required to have the intuitionistic views, interactive mechanism and flexibility. Actually, some ADLs can provide the tools of modeling and compiling, e.g. Rapide [8], DARWIN [9], and XYZ/E [10] [11]etc. But these ADLs are based on object-oriented technology, and are difficult to support the design process of cooperation-oriented system. But the agent-oriented program and compiler platform are still not satisfied with the real application of business system.

It is no doubt that the ADLs and their tools are still very helpful for architecture design. But the reuse of design philosophy cannot depend only on the formal language because any popular description language has its limitations. For example, the ADLs based on component are hard to describe the applications of multi-agent system or other distributed cooperation system. After all, the architecture design should focus on such issues as: cooperation, competition of resource and coordination of conflict etc. Therefore, we employ the cooperation as the element of modeling of SA, which is an efficient approach for such application systems. The cooperation is the integration unit of the local tasks and responsibilities and the entity of analysis and design; therefore, the knowledge and experience of architecture design can be wholly represented and transferred in this model of SA. As the specialists of domain analysis need not be familiar with program language and tools, the prototype of system architecture can be described independently. Here, we insist that the architecture descriptions should be unrelated to the real implemental tools or program language; they should keep such key points: tasks of system, performance goals, security, and feasibility.

### **3 Architecture Description of Cooperation-Oriented System**

In above analysis, we discussed the description of architecture, and argued that the formal method is preferred. Although the standard of IEEE on software architecture did not designate the element of modeling, the component is still the primary choice for most of the ADLs. These ADLs based on component can be classified into two languages: IIL (Implementation Independent Language) and ICL (Implementation Constraining Language) [18]. The former languages have WRIGHT [12] and Rapide; Unicon and MetaH [13], and XYZ/E belong to the ICL. ADL should have the complete and sound syntax and semantic support. A mature ADL should provide such basic services as: architecture configuration, style, architecture composite, properties

analysis etc. Here, we choose the traditional one of ADLs-WRIGHT as the basic description language of cooperation-oriented architecture. We extended the notions of WRIGHT, and the semantic description is still based on CSP. We called this extended WRIGHT as WRIGHT\* in order to differ from the WRIGHT.

As to the syntax of WRIGHT\*, we just added some notions into the traditional framework. For example, as to the element of architecture, we added the *cooperation* and *supporter* into the set of basic elements. The definitions of *Cooperation* and *Supporter* are related to the definition 2 and definition 3. In order to describe the uniform pattern of behavior in the architecture style, we extended the concept of *Interface Type* to support such added properties as: *Task*, *Agent* and *Service*, and added the operators of set related to *Cooperation* and *Supporter*: *Tasks()*, *Agents()*, *Services()* and *Customers()*. In WRIGHT\*, the agent is taken as the meta-element of modeling, just like the component enconces the concept of object of OO method.

In the case of process algebra, e.g. CSP, PEAP and CCS etc, the process is described as entity, even agent in some situations. In fact, process algebra embodies the operator of synchronization to describe the concurrent. Here, we extended the meaning of synchronization to cooperation including more behaviors, e.g. intercourse, communication, negotiation, task assignment etc. Therefore the cooperation is handled to be a super-event or a composite event. We induct the symbol  $\Theta$  as the cooperative operator to represent the cooperation among several processes (agents).

**Definition 5.** The cooperation event is a composite event and represents the several processes (agents) performing the related behaviors to attain the common goals of system.

Let  $P$  is the set of processes,  $\text{Event}_{\text{coop}} = \forall p_i, p_j \in P \ i, j = 1 \dots n, i \neq j, (p_i \Theta p_j)$  is a composite event; and the operator  $\Theta$  has such characters:

- 1) **Atomicity.** This character is to stress that the event of cooperation can not be divided into an integration of fragment events.
- 2) **Terminable.** if  $i=j$ , then  $(p_i \Theta p_j) = (p_i \parallel p_j) \square \sqrt{\phantom{x}}$ , where  $p_i, p_j \in P$
- 3) **Symmetry.**  $\forall p_i, p_j \in P \ i, j = 1 \dots n, i \neq j, (p_i \Theta p_j) = (p_j \Theta p_i)$
- 4) **Goal-consistency.**  $\forall p_i, p_j, p_k \in P \ i, j, k = 1 \dots n, i \neq j \neq k$ , if the goal of  $(p_i \Theta p_j)$  is the same as the event  $(p_j \Theta p_k)$ , then  $(p_i \Theta p_j) \approx (p_j \Theta p_k)$ . It means that these two events of cooperation have the same effect on the next event, and then these two events can be equivalence. In reverse, the conclusion is tenable. This character is important for the higher composite of events.

## 4 Conclusions

Aimed at the cooperation-oriented system, we brought forward a novel model of architecture in which the cooperation is the basic element of modeling. To describe this novel model, we extended the traditional WRIGH language. With the development of architecture description, the traditional model of architecture based on component is limited to describe those cooperative systems. Concerning the complex cooperative system, the object-oriented philosophy and method have a little awkward and trivial on elaborating the aggregation of agents. Cooperation is taken as the

element of architecture modeling which is coincided with the development of architecture description theory. Just like the software architecture based on technology of component represents the evolution of object-oriented technology, the cooperation-oriented system can represent the future development of agent-oriented technology. The cooperation is the encapsulation of tasks and goals of organization, and this philosophy can improve the architecture design for complex cooperative system, and promote the reuse of architecture framework.

To describe the cooperation-oriented architecture, we tailored the WRIGHT, and called this novel ADL as WRIGHT\*. In objective, WRIGHT\* based on CSP still has the limitations in describing the interaction with time factor. Our next works will include such aspects as: 1) further formal description of cooperation-oriented system, 2) the completeness and soundness of WRIGHT\*, 3) system design of cooperation-oriented system, 4) design pattern of cooperation-oriented system and so on.

## References

1. Mary Shaw, David Garlan. Software architecture: perspectives on an emerging discipline. Prentice Hall, (2004).
2. Sun Chang-ai, Jin Mao-zhong, Liu Chao. Overviews on Software Architecture Research. Journal of Software, Vol.13 (2002) 1228-1237.
3. Ma Jun-tao, Fu Shao-yong, Liu, Ji-ren. A-ADL: an ADL for multi-agent system. Journal of Software, Vol.11 (2000) :1382-1389
4. Wooldridge M, Jennings N. Intelligent Agents: theory and practice. The knowledge Engineering Review, Vol.10 (1995) 115-152.
5. Zhang Jie, Gao Liang, Li Pei-gen. Application on advanced manufacture of multi-agent technology. Science Press Beijing (2004).
6. Woolridge M, Jennings N R. Towards a theory of cooperative problem solving. In Proc. Modeling Autonomous Agents in a Multi-Agent World (MAAMAW-94), pages 15-26, Odense, Denmark (1994).
7. IEEE ARG. IEEE's Recommended Practice for Architectural Description, IEEE P1471-2000 (2000).
8. Luckham DC, Vera J. An event-based architecture definition language. IEEE Transactions on Software Engineering, Vol.21 (1995):717-734.
9. Magee, J., Kramer, J. Dynamic structure in software architectures. In: Kaiser, G.E., ed. Proceedings of the ACM SIGSOFT'96: the 4th Symposium, Foundations of Software Engineering (FSE4). New York: ACM Press (1996) 3-14.
10. Tang Zhi-song. The Goal, Meaning, Effect and Application of the XYZ System. Journal of Software, Vol.10 (1999) 337-341.
11. Zhu Xue-yang, Tang Zhi-song. A Temporal Logic-Based Software Architecture Description Language XYZ/ADL. Journal of Software, Vol.14 (2003) 713-720.
12. ALLEN, R. J. A formal approach to software architecture. Ph.D. thesis, School of Computer Science, Carnegie Mellon University, Pittsburgh, PA., May (1997).
13. Medvidovic N , Taylor RN. A classification and comparison framework for software architecture description language. IEEE Transactions on Software Engineering, Vol.26 (2000) 70-93.

# An A-Team Based Architecture for Constraint Programming

Yujun Zheng<sup>1,2</sup>, Lianlai Wang<sup>1</sup>, and Jinyun Xue<sup>2,3</sup>

<sup>1</sup> Systems Engineering Institute of Engineer Equipment, 100093 Beijing, China  
uchengz@yahoo.com.cn

<sup>2</sup> Institute of Software, Chinese Academy of Sciences, 100080 Beijing, China

<sup>3</sup> College of Computer Information and Engineering, Jiangxi Normal University,  
330027 Nanchang, China

**Abstract.** The paper proposes an agent-based constraint programming architecture that we have successfully applied to solve large, particularly combinatorial, operations problems. The architecture is based on the asynchronous team (A-Team) in which multiple problem solving agents cooperate with each other by exchanging results to produce a set of non-dominated solutions. We extend the A-Team by introducing CSP-specific agents, explicitly defining solution states, and enabling solution decomposition/composition, and thereby improve the performance, reliability, and automation of constraint programming significantly.

## 1 Introduction

Constraint programming is a widely used paradigm for declarative description and effective solving of constraint satisfaction problems (CSP)[1]. Today's real-world problems increasingly involve complex sets of objectives and constraints, and traditional approaches typically result in a large monolithic model that is difficult to solve, understand, and maintain. In recent years, constraint programming has integrated many artificial intelligence (AI) techniques to enhance its capability to represent and automatically enforce diverse and complex constraints inherent in large, real-world applications [2]. Many agent-based architectures for solving multi-objective optimization-problems have been proposed (e.g., [3,4]). Focusing on developing a solution method that allows each agent to be specialized to a single constraint or objective, [5] proposes the asynchronous team (A-Team), an agent framework in which an asynchronous team of agents shares a population of solutions and evolves an optimized set of solutions.

We propose here a multi-agent constraint programming architecture that extends the A-Team by introducing specific agents for CSP specification construction and constraint relaxation, and explicitly defining different CSP solution states during the problem solving process. The architecture has been implemented with a prototype tool in our integrated development environment [6] and applied to solve real-world combinatorial planning and scheduling problems.

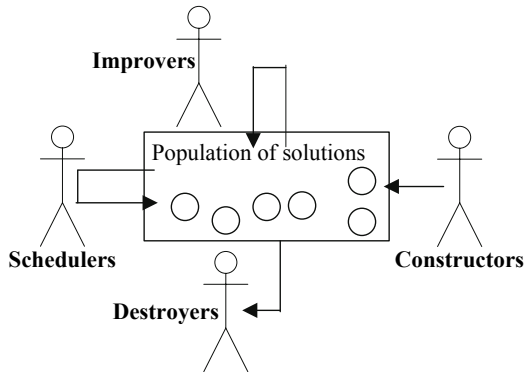
The remaining of the paper is structured as follows: Section 2 briefly introduces the preliminaries of CSP and A-Team. Section 3 describes our architecture and its typical problem solving process, and Section 4 concludes with discussion.

## 2 Preliminaries

An instance of the CSP (see [7] for the formal definition) is described by a set of variables, a set of possible values for each variable, and a set of constraints between the variables. In [8] we also formally define aggregation, normal combination, and orthogonal combination of CSPs, on top of which problems can be specified, proved correct, and composed together in such a way to preserve their intra-properties as well as inter-relations.

A-Team is an agent based architecture in which multiple problem solving methods (agents) cooperate to produce a set of non-dominated solutions. As shown in Fig. 1, an A-Team consists a population of candidate solutions and four types of agents:

- *Scheduler*: the users that act as agents by defining problems or modifying existing solutions.
- *Constructors*: the agents that take problems and create initial solutions.
- *Improvers*: the agents that take existing solutions from the population and modify them to produce better solutions.
- *Destroyers*: the agents that remove low quality or redundant solutions from the population and keep the size of the population in check.



**Fig. 1.** A-Team framework

Software agents in an A-Team are composed of four components that enable them to participate into the teams and assist in the population evolution:

- A requester that requests notification of system events and determines if the agent has become relevant to the current situation.
- A selector that picks zero or more solutions as input to the optimization operator.
- An operator that runs and produces zero or more result solutions and passes them to the distributor.

- A distributor that filters the results and adds them to the output population.

The selection of agents to run and some of the algorithms used by the agents are probabilistic. Although the results can not be guaranteed theoretically, empirical results suggest that this approach produces [9].

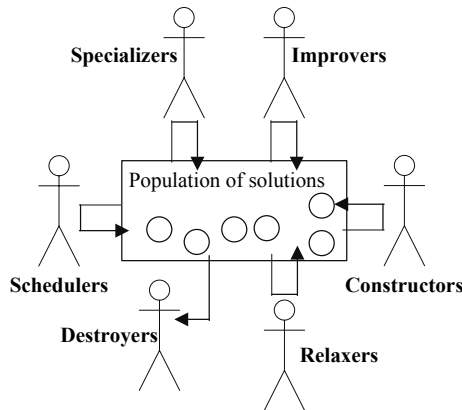
### 3 A-Team Based Constraint Programming Architecture

A-Team provides an intrinsically modular, scalable and robust computing environment for constraint programming, but it still subjects to the following special limitations:

- It is difficult and time-consuming for schedulers (human agents) to formally define initial problems in terms of CSP.
- It is inefficient in A-Team to define, improve and combine heuristics for problem solving.
- Most resulting solutions could not give enough information about the trade-offs between the objectives and constraints.

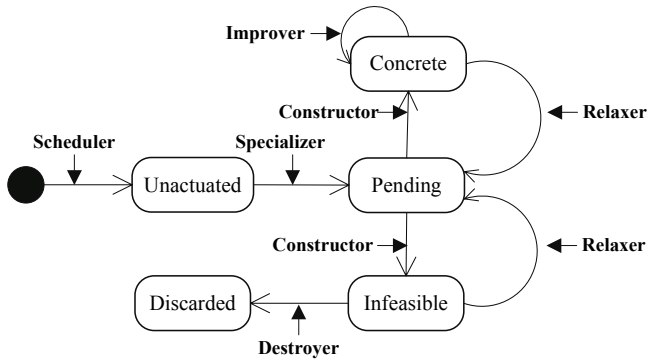
To improve its effectiveness and intelligence for constraint programming, we extend A-Team by introducing the following two types of agents (see Fig. 2):

- *Specializers*: the agents that build CSP specifications for empty solutions.
- *Relaxers*: the agents that take infeasible, objective-sensitive, constraint-sensitive, or over-complicated solutions from the population and relax some constraints for the problem.



**Fig. 2.** The A-Team based constraint programming architecture

Our architecture is specifically designed for solving complex multidisciplinary problems in constraint programming. That is, different specializers build sub-specifications of the CSP according to their domain-specific knowledge, and the



**Fig. 3.** The state diagram of CSP solutions

constructors use their specific algorithms to generate initial sub-solutions according to separated specifications. When all the sub-problems are successfully (Pareto-optimally) solved, their solutions are synthesized to a full, optimized solution. We further define five states for a CSP solution during the problem solving process, as illustrated in Fig. 3.

### 3.1 Specification Construction

First, a scheduler creates an initial problem that is composed of objectives, variables and their domains, and constraints, and an *unacted solution* is generated and sent to the solution space (population). Taking an unacted solution from the population, the specializer extracts the part of the problem whose objectives and constraints fall into its domain category. It then uses the domain-specific knowledge to build a CSP specification for the problem part, writes the sub-specification into the solution, and sends it back to the population. Here we call a problem part together with its sub-specification as a *pending sub-solution*. The specializers of other domain categories continually build sub-specifications for the solution until all parts of the initial problem is built into pending sub-solutions, that is, the whole unacted solution becomes a *pending solution*.

### 3.2 Problem Solving

A constructor takes a pending solution from the population, traverses all the pending sub-solutions and picks out those whose CSP specifications are subject to the problem-solving tactics of the constructor. The constructor then tries one of his concrete algorithms to solve the sub-CSP; if successful, the pending sub-solution becomes a *concrete sub-solution*. Constructors with other tactics continually try their algorithms to solve pending sub-problems remaining in the solution. If all pending sub-solutions become concrete, the whole pending

solution itself becomes a *concrete solution*. After being passed by the constructors with all of the solving statics, if the solution still has any pending sub-solution, it becomes an *infeasible solution*.

### 3.3 Solution Improving and Relaxing

There are two types of improvers in our architecture: one takes single concrete sub-solution as input and try to improve it in one or more objectives; the other reallocates the resource between sub-solutions and try to improve one or more objectives for the whole solution. The global optimization is achieved mainly by constraint propagation [10], feasibility reasoning and optimality reasoning [11].

When the scheduler proposes an initial problem, he may explicitly add a mark *RelaxLevel* (ranging from 0 to 2) to the unactuated solution that states the level of permitted constraint relaxation:

- For an infeasible solution whose *RelaxLevel* is 1 or 2, the relaxer heuristically violates some problem constraints based on the constraint sensitivity analysis, increases its *RelaxCount*, and sends it back to the population. The solution and all its relaxed sub-solutions become pending.
- For a concrete solution whose *RelaxLevel* is 2, the relaxer tries to violate some problem constraints and tests the result: if some slight violations yield significant improvement on one or more objectives, increases its *RelaxCount*, and sends it back to the population; otherwise remains the solution unchanged.
- For a concrete solution whose *RelaxLevel* is 1 and whose *RelaxCount* is not zero, the improver tries to remove the constraint violations through iterative repair [12].
- For an infeasible solution whose *RelaxLevel* is 0, or whose *RelaxCount* exceeds a per-defined limit, the Destroyer removes it from the population, that is, the solution is discarded.

## 4 Conclusion

Constraint programming is capable of handling a wide range of optimization problems in the areas of planning and scheduling. In combination with agent techniques, its capability to represent complex constraints and its intelligence and effectiveness in problem solving can be improved tremendously.

The paper reports an multi-agent architecture for solving large, particularly combinatorial, CSPs. It is based on A-Team in which multiple problem solving agents cooperate with each other by exchanging results to produce a set of non-dominated solutions. By introducing two types of CSP-specific agents, namely specilizer and relaxer, the architecture enables solution composition and state transformation, and therefore contributes greatly to the concern separation and performance improvement. Ongoing efforts include investigating learning and other adaptive behaviors of the agents in the architecture.



## References

1. Bistarelli, S.: Semirings for Soft Constraint Solving and Programming. Lecture Notes in Computer Sciences, vol. 2962 (2004) 1-20
2. Alguire, K.M. and Gomes, C.P.: Technology for Planning and Scheduling under Complex Constraints. Proceedings of Society for Optical Engineering, Boston, USA (1997) 101-107
3. Beck, J.C. and Fox, M.S.: Supply chain coordination via mediated constraint relaxation. Proceedings of 1st Canadian Workshop on Distributed Artificial Intelligence, Banff, AB (1994) 61-72
4. Liu J.S., and Sycara, K.P.: Multiagent coordination in tightly coupled task scheduling. Proceedings of 2nd International Conference on Multiagent Systems, Kyoto, Japan (1996) 164-171
5. Talukdar, S.N., Baerentzen, L., Gove, A., and Souza, P.D.: Asynchronous Teams: Cooperation Schemes for Autonomous Agents. Journal of Heuristics, vol. 4, no. 4 (1998) 295-321
6. Zheng, Y.J. and Xue, J.Y.: MISCE: A Semi-Automatic Development Environment for Logistic Information Systems. Proceedings of IEEE International Conference on Service Operations and Logistics, and Informatics, Beijing, China (2005) 1020-1025
7. Tsang, E.P.K.: Foundations of Constraint Satisfaction. Academic Press, London, UK (1993)
8. Zheng Y.J., Wang L.L., and Xue J.Y.: A Constraint Programming Framework for Integrated Materiel Logistic Support. Journal of Nanjing University, vol. 40, no. 10 (2005) 30-35
9. Akkiraju, R., Keskinocak, P., Murthy, S., and Frederick, W.: An Agent-Based Approach for Scheduling Multiple Machines. Applied Intelligence vol. 14 (2001) 135-144
10. Pepper, P. and Smith, D.R.: A High-level Derivation of Global Search Algorithms (with Constraint Propagation). Science of Computer Programming. vol. 28, special issue on FMTA (Formal Methods: Theory and Applications) (1996) 247-271
11. Focacci, F., Lodi A., and Milano, M.: Optimization-Oriented Global Constraints. Constraints, vol. 7 (2002) 351-365
12. Zweben, M., Daun, B., Davis, E., and Deale, M.: Scheduling and rescheduling with iterative repair. Intelligent Scheduling (eds. Zweben, M. and Fox, M.S.), Morgan Kaufman (1994) 241-255

# The Efficient and Low Load Range Queries in P2P\*

Shui Chao, Zhou Pen, Jia Yan, and Zhou Bing

School of Computer, National University of Defense Technology,  
410073 Changsha, China

**Abstract.** To enable efficient and appropriate uses of the resources in web, it is important to provide range query to keep track of the availability and attributes of millions of service which are geographically distributed. Range query in P2P face a performance tradeoff problem between the efficiency of query and number of message: Optimizing one tends to put pressure on the others. In this paper, a range query algorithm had proposed which lookup for every node in range can be done via  $O(\log N)$  hops, and number of messages is trend to  $O(\log N)+m-1$ . We evaluate our system in this paper via a simulation and show that its design along with particular Range-query algorithm meets the goals of efficient query and low message load.

## 1 Introduction

The P2P which base on the DHT facilitates the capacity of exact match for Service discover: system can locate the service presented in overlay network via  $O(\log N)$  hops. But the need for range query had increased recently, such as search for service which provide storage capacity and  $100M < \text{size} < 200M$ , or for the service whose price is less than 100\$.

The performance of range query in DHT is related to the size of query range, which including the efficiency of lookup and load of message. Some research [2] point out that the lower bound of lookup efficiency is  $O(\log N)$ , and the lower bound of message load is  $O(\log N)+m-1$ , where  $N$  is number of nodes in system and  $m$  is the size of query range. Recently, some [2] range query techniques had been proposed which find out all resource in range via  $O(\log N)$  hops but the message load is high, some technique [4] achieve the lower load but inefficient lookup. It shows optimize one performance tends to put pressure on the others. The tradeoff of the two performances is a challenge in range query.

In this paper, we propose a range query algorithm base on the Cactus topology, whose efficiency of lookup and load are tends to the lower bound discussed above. Especially in the large scale of nodes in P2P system, the range-query in Cactus characterizes efficient lookup and lower load of message. To our knowledge, the range query in Cactus is the first one who solves the tradeoff problem between the lookup efficiency and message load in constant-degree system.

---

\* Supported by the National Basic Research Program of under Grant Nos. 2005cb231804.

## 2 Range Query Algorithm

### 2.1 The Topology of Cactus

The Cactus[11] is base on the CCC and 2-tree. Each node is represented by a pair of indices  $(k, a_{d-1}a_{d-2} \dots a_0)$ , where  $k$  is a cyclic index and  $a_{d-1}a_{d-2} \dots a_0$  is a cubical index. The cyclic index is an integer, ranging from 0 to  $d-1$  and the cubical index is a binary number between 0 and  $2^{d-1}$ . Each node in Cactus has 6 neighbors: two ring neighbors in CCC cycle, one cubic neighbor, one farther and tow children in tree. Each node graph has three neighbors in CCC: one cubical neighbor whose coordination is  $(k, a_{d-1}a_{d-2} \dots \bar{a}_k \dots a_0)$ ; two cyclic neighbors, one's coordination is  $((k-1) \bmod d, a_{d-1}a_{d-2} \dots a_0)$ , and another is  $((k+1) \bmod d, a_{d-1}a_{d-2} \dots a_0)$ . Each node  $(0\dots 01a_k, \dots a_0)$  has two children, the coordination of the left child is  $(0\dots 011a_k, \dots, a_0)$ , and the coordination of the right child is  $(0\dots 010a_k, \dots, a_0)$ .

In paper [11], we had shown that the mean length of routing path is  $(5/4)*d$ . Since  $d$  is less than the  $\log N$ , so the lookup efficient of Cactus achieve the mean hops about  $\log N$ .

### 2.2 Key Assignment

For “continuous” types of attributes, such as processing or storage capacity, querying of attribute ranges is useful: first, finding resources with an approximate attribute value might be sufficient; furthermore, we might want to locate all resources with attribute values larger (or smaller) than a certain number.

The size of ID space in  $d$ -dimension Cactus is  $d*(2^d-1)$ . Supports the attribute value varies from  $n_1$  to  $n_2$ , and we divide the attribute range into  $d*(2^d-1)$  child range:

- (1) divide the  $[n_1, n_2]$  into  $2^d-1$  ranges named cube-range, and the  $i^{\text{th}}$  cube-range present with  $(i_{d-1} \dots i_0)$ ;
- (2) divide the one cube-range into  $d$  ranges named ring-range, and present with a integer varies from 0 to  $d-1$ .

So every attribute value  $i$  in  $[n_1, n_2]$  can present with coordinate (ring\_range, cube\_range), that is

$$a_{d-1}, \dots, a_0 = \lfloor (i + Value) * 2^d / m \rfloor \tag{1}$$

$$k = \left\lfloor \frac{value * 2^d}{m} \right\rfloor \tag{2}$$

For a given attribute value  $i$ , the Cactus maps it to node coordinate following formula (1)(2), and cube-range index is the cubical index and ring-range index is cyclic index. For example, the attribute value 118 should store in the node (1,1101), or store in (2, 1101) when the node(1,1101) doesn't participant, or store in node(1,0101) when no node with cubic index (1101) existing in network.

### 2.3 Range Query

Support the attribute value varies from  $n_1$  to  $n_2$ , and query range varies from  $m_1$  to  $m_2$ , then there are  $m = \left\lceil \frac{n_2 - n_1}{(m_2 - m_1)/2^d} \right\rceil * d$  nodes should be search in Cactus. The

simplest range query algorithm is flooding: send query to every node in the range, and its time complexity is  $O(d)$  in Cactus but the mean message number is  $5/4 * d * m$ . It is too high load for network when  $m$  increasing. For decrease the load and keep the lookup efficiency, a search tree had proposed as follow:

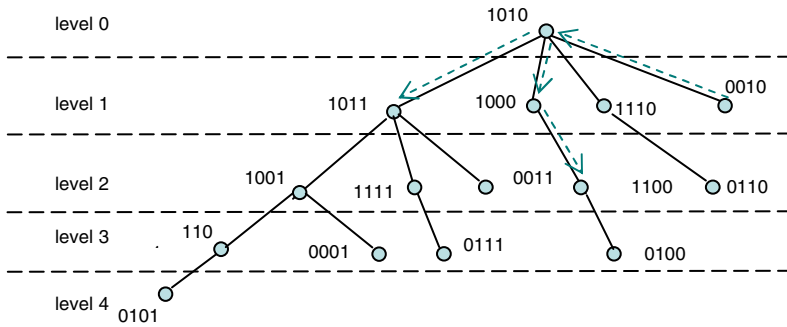


Fig. 1. The search tree with root(1010)

- (1) Select a node as root which locates in the range.
- (2) There are at most  $(d-i-j)$  children for the node which locate on the  $i^{th}$  level of tree and the  $j^{th}$  position in this level.
- (3) the index of the  $k^{th}$  child for each node  $a(a_{d-1}, a_{d-2}, \dots, a_0)$  is  $(a_{d-1}, \dots, \bar{a}_k, \dots, a_0)$

For example, the search tree is show in Fig 1 where the root is  $a(1010)$ .  $a(1010)$  have 4 children since it is in the level 0, and  $b(1011)$  is its first child since the  $0^{th}$  bit of  $b$  is contrary to the  $0^{th}$  bit of  $a$ . Please notice that the node (1000) only have one child since there is no node (0000) existing in Cactus.

Support the requester is root of search tree and use the flooding algorithm in search tree, some routing for different node may have common path. For example, the routing path from root (1010) to node 0001 and 0101 have common path  $1010 \rightarrow 1011 \rightarrow 1001$ . If the range query message send to node 1001 firstly, then it resend message to node 0001 and 0101 respectively, it will decrease the number of message in flooding algorithm and keep the lookup efficiency. Base on this idea, we design a range query algorithm named smart-broadcast. Support the current node is  $p(k, p_{d-1} \dots p_0)$ , the smart-broadcast present as following:

- (1) if  $p$  is in the query range, return  $p$  to requester
- (2)  $p$  creates set  $S_i, 0 < i < d$ . For each node in query range, if the minimal difference bit(MDB) between the node's cubic index and  $p$ 's cubic index is  $\gamma$ , then add this node to set  $S_\gamma$ . that is

$$S_i = \{(x, b_{d-1}, b_{d-2}, \dots, b_0) \mid (x, b_{d-1}, b_{d-2}, \dots, b_0) \in [m_1, m_2], b_i \neq p_i, b_j = p_j (j < i)\} \quad (3)$$

- (3) if the  $S_k$  is not empty, then forwards query range message to  $p$ 's cubic neighbor, the message describe as  $\langle \text{requesterIP}, \text{queryRange} \rangle$ . The requesterIP is the requester's IP address, and queryRange is the  $S_k$ .
- (4) support the successor neighbor's cyclic index is  $j$ , then merge the  $S_i$  into one set  $Q$  where  $i$  varies from  $j$  to  $d$ , and forwards the message  $\langle \text{requesterIP}, \text{queryRange} \rangle$  to successor. The queryRange is  $Q$ .
- (5) merge the  $S_i$  into a set  $\beta$  where  $i$  varies from  $k+1$  to  $j-1$ .
- (6) divide  $\beta$  into two set. Each node in first set is the child of  $p$ , another has nodes else.
- (7) forwards first set to child neighbor, and forwards second set to farther neighbor.

There is one question we must answer that who is the root of search tree. We use Nearest Node As Root(NR) strategies to solve this problem: the node who is locate in the query range, whose cubic index is closest to requester's cubic index, and whose cyclic index is 0.

For example, support the attribute value varies from 80 to 93 in 4-dimension Cactus, and then cube-range in the query range varies from 10 to 12. Using the NR strategy, the requester forwards message to the node(0,1010), who is the closest to requester and locate in the query range. The node(0,1010) knows itself is in the query range, and return the resource information storing in it. Then the node creates 4 empty sets following the formulate 3, and get 2 sets: {11}, {12}, or {1101}, {1100}. So node send query message to (0,1011) and (1,1010) respectively. Following the smart-broadcast algorithm, the query will reach all nodes that are located in query range. The routing path is show in fig 2 as the dotted line.

### 3 Evaluation

We compare smart-broadcast algorithm to other range query algorithm including DCF, Armada[2], and Mercury[3]. Since DCF base on CAN system, we set the CAN's dimension is 3. The Mercury is base on the small-world model, and we set the number of long-link in Mercury is 4 and random-link is 2. So each node due to different DHT has 6 neighbors.

The experiments evaluate the lookup efficiency firstly, and condition is same as the 4.1. The mean lookup path length is shown in fig 2(a). The fig 2(a) plots that mean hops of DCF and Mercury increases quickly with the increasing size of nodes, but the mean hops of Armada and smart-broad increases slowly which show as exponential curve. The time complexity for Armada is  $O(\log_2 N)$ , and smart-broad is  $O(d)$  which is about  $3/2 * \log_2 N$  as result in Fig 2(a). The reason that the smart-broadcast is slower than Armada in lookup efficiency is because that MR strategy must locate root at first.

On the other hand, the lookup efficiency with the increasing size of query range is evaluated in next experiments. The size of query range varies from 10 to 100 which increase 10 each time. The mean lookup hops are showed in Fig 2(b). The Fig 2(b) plots that the size of hops does not increase for Armada and smart-broadcast when the size of query range increases, because the upper bound of lookup efficiency is  $O(\log N)$ .

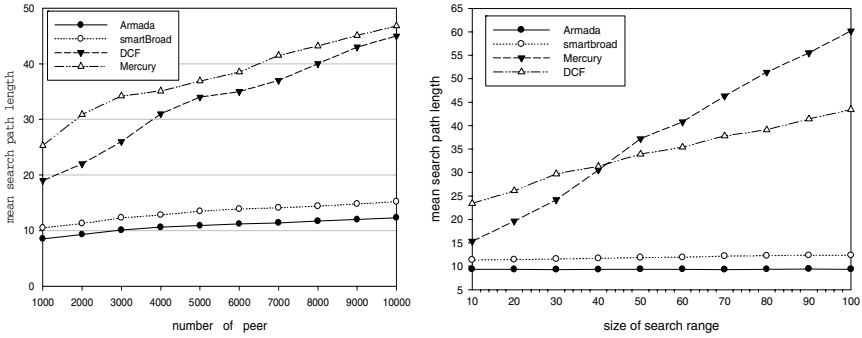


Fig. 2. The lookup efficiency due to different DHTs

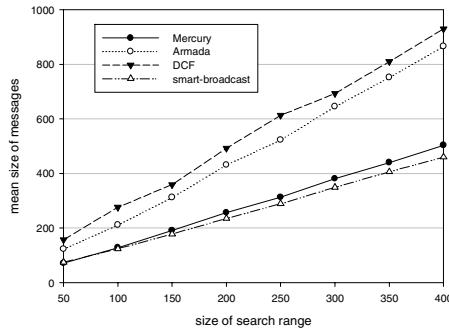


Fig. 3. The message load due to different DHTs

For evaluate the message load due to different DHTs, the size of query range varies from 50 to 400 which increase 50 each time, and there are 2000 nodes in network. Due to different scale, the mean message load for 1000 experiments is shown in Fig 3. The size message load of all algorithms increases as linearity curve due to increasing size of query range. But the size of message load for Armada and DCF is 2 times than smart-broadcast. While we prove the upper bound of number of message is  $O(d)+2m$  following smart-broadcast algorithm, but average number of messages is tend to  $O(\log N)+m$  in simulation.

## 5 Conclusion

In this paper, we discuss the tradeoff problem between the lookup efficiency and message load. A constant-degree P2P system named Cactus and a range query algorithm had proposed in this paper. Every node can be reached in  $O(d)$  hops and mean size of message trend to  $O(\log N)+m-1$ , where  $m$  is the size of query range. Three different strategies had proposed for different performance, the RR strategy for minimal message load, NR for best lookup efficiency, and MR is better tradeoff

performance than RR and NR. To our knowledge, the smart-broadcast is the first range query algorithm which achieves the lower bound of lookup efficiency and message load in constant-degree system. Lookup resource with multiple attribute is the future work we focus on.

## References

- [1] Jun Xu, Abhishek Kumar, Xingxing Yu. On the Fundamental Tradeoffs between Routing Table Size and Network Diameter In peer-to-peer Networks. *IEEE Journal on Selected Areas in Communications (JSAC)*, Vol. 22, No. 1, January 2004.
- [2] Dongsheng Li, Xicheng Lu, et al. Topology and Resource Discovery in Peer-to-Peer Overlay Networks. *Proceedings of International Workshop on Storage Grid and Technologies (SGT'2004)*, Lecture Notes in Computer Science 3252, Springer, 2004
- [3] Ashwin R. Bhambe, Mukesh Agrawal, Srinivasan Seshan. Mercury : Supporting Scalable Multi-attribute Range Queries. *Proc. of SIGCOMM'04*, Portland, Oregon, USA, 2004.
- [4] D. Oppenheimer, J. Albrecht, D. Patterson, and A. Vahdat. Distributed Resource Discovery on Planetlab with SWORD. *Proc. of the First Workshop on Real Large Distributed Systems (WORLDS'04)*, Santa Fe, New Mexico , USA, December 2004.
- [5] M. Cai, M. Frank, J. Chen, and P. Szekely. MAAN: A Multi-attribute Addressable Network for Grid Information Services. *Proc. of the 4th International Workshop on Grid Computing (Grid'2003)*, Phoenix, AZ, USA, 2003.
- [6] Adina Crainiceanu Prakash Linga Johannes Gehrke Jayavel Shanmugasundaram. PTree: A P2P Index for Resource Discovery Applications. *Proc. of WWW2004*, New York, USA, May 2004.
- [7] Hamid Nazerzadeh and Mohammad Ghodsi. RAQ: A Range-Queriable Distributed Data Structure. *Proc. of 31st Annual Conference on Current Trends in Theory and Practice of Informatics (SOFSEM'05)*, Liptovsky Jan , Slovak Republic, LNCS 3381, pp. 269-277, 2005.
- [8] S. Milgram, "The small world problem," *Psychology Today* 1, 61 (1967).
- [9] Yatin Chawathe, Sriram Ramabhadran, Sylvia Ratnasamy, Anthony LaMarca, Scott Shenker, Joseph Hellerstein. A Case Study in Building Layered DHT Applications. *Proc. of SIGCOMM'05*, Philadelphia, Pennsylvania, USA, August 2005.
- [10] Artur Andrzejak and Zhichen Xu. Scalable, Efficient Range Queries for Grid Information Services. *Proc. of the Second IEEE International Conference on Peer-to-Peer Computing (P2P'2002)*, Linköping, Sweden, September 2002.
- [11] ShuiChao, et al., *Cactus: a new constant-degree and fault tolerate P2P overlay*. the ninth Pacific Rim International workshop on Multi-Agents, Guilin, China, 2006.

# Research of Agent Based Multiple-Granularity Load Balancing Middleware for Service-Oriented Computing\*

Jun Wang, Di Zheng, Quan-Yuan Wu, and Yan Jia

School of Computer Science,  
National University of Defence Technology,  
Changsha, Hunan, China 410073  
junwang@nudt.edu.cn

**Abstract.** With the rapid development of computer technology, the distributed applications scale up increasingly. Web service becomes more useful, and more software systems begin to make use of service-oriented architecture SOA. To improve the dependability and scalability of SOA, one effective way is to provide service replicas and balance loads among the replicas via adaptive load balancing service based on the middleware. By using middleware, we can satisfy the urgent demands of performance, scalability and availability in current distributed service-oriented applications. However, most of the current load-balancing middleware adopt the per-replica load monitoring granularity, and if multiple kinds of service groups are deployed to the same host problems will arise such as redundant load monitoring and weak scalability. To solve these problems, we design and implement a multiple-granularity load balancing middleware model by using agents. In this paper, we will present the architecture of our model with the simulation results.

## 1 Introduction

To service many increasing online clients transmitting a large number of busy requests and provide dependable services with high quality constantly, we must make the distributed computing systems more scalable and dependable. And even under high load, the systems must still support the services as usual. So, effective load balancing mechanisms must be used to distribute the client workload equitably among back servers to improve responsiveness. As we known, load balancing mechanisms can be provided in any or all of the following layers in a distributed system:

- **Network-based load balancing:** This type of load balancing is provided by IP routers and DNS. However, load balancing at this layer is somewhat limited by the fact that they do not take into account the content of the client requests.
- **OS-based load balancing:** This type of load balancing is provided by distributed operating systems via *load sharing*, and *process migration* [1] mechanisms. Load balancing can be achieved via *process migration* mechanisms [2], where the state of a process is transferred between nodes.

---

\* This work was funded by the National Grand Fundamental Research 973 Program of China under Grant No.2005cb321804, the National High-Tech Research and Development Plan of China under Grant No.2004AA112020.



- **Application-based load balancing:** This type of load balancing always has close relation with the applications, such as in [3] the workload is distributed based on graph to avoid the resource bottleneck of the centric systems. However, the cost for balancing load in this way is too high and not general.
- **Middleware-based load balancing:** This type of load balancing is performed in middleware, often on a per-session or a per-request basis. The key enterprise applications of the moment all make use of the middleware based distributed software systems to handle complex distributed applications. The advantages of middleware can make it easy to provide load balancing for the cluster and manage the kinds of redundant services effectively.

There are different realizations of load balancing middleware. For example, stateless distributed applications usually balance the workload with the help of naming service [7]. But this scheme of load balancing just support static non-adaptive load balancing and can't meet the need of complex distributed applications. For more complex applications, the adaptive load balancing schema [4] [5] [6] is needed to take into account the load condition dynamically and avoid override in some node.

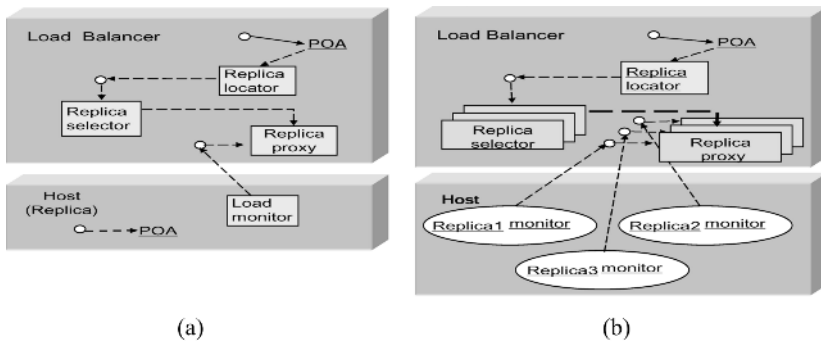


Fig. 1. (a) Single Replica in One Host (b) Multiple Kinds of Replicas in One Host

As depicted in figure 1(a), in existing adaptive load balancing realizations, there is only one kind of service replica in one host and the load monitoring granularity is per-replica (host). However, with the growth of computing, users begin to deploy more services in each host and more load-balancing systems look like figure 1(b). These service replicas which have their own load monitors will bring forth some new problems to the traditional per-replica (host) granularity load balancing middleware.

- **Redundant load monitoring:** For the different replicas residing in the same host, their load metrics will be similar to some degree. Even in some extreme situations, all the replicas may have the same load metric. So, in the traditional per-replica (host) granularity load balancing middleware, all the independent load monitors execute the similar load monitoring frequently and redundantly. This redundancy will add unnecessary overhead to the load balancing systems.
- **Weak scalability:** To some certain load metric, adding new replica will affect existing groups. For example, supposing there have been several replicas which

all take care of the CPU utilization. If adding a new replica M1 which has the same load metric, then the selection of M1 will heaven the CPU utilization and the load information of the other replicas will invalidate. In many situations, this invalidate will lead to load unbalancing. To avoid this load unbalancing, we must change the load monitoring recycle and this will add more unnecessary and redundant overhead.

So, in this paper we design and implement an agent based load balancing middleware. At the same time, because CORBA [9] has become one of the mainstream technologies for the distributed development. In this paper, we will focus on CORBA load balancing middleware.

## 2 Agent Based Multiple-Granularity Load Balancing Model

### 2.1 Agent Based Virtual Group

Most of the replicas may pay their attention to the common information such as CPU utilization, memory utilization and so on. From figure 1 we can see in traditional per-replica (host) granularity middleware each load monitor completes its monitoring independently and this will lead to redundant monitoring. So, we organize the replicas having the same load metric by bring in some agents. As depicted in figure 2, supposing the service groups G1, G2, G3 pay the same attention to CPU utilization and according replicas G1-R1, G2-R1, G3-R1 reside in the same host. After the replicas register with the agent, the above replicas will form a virtual group.

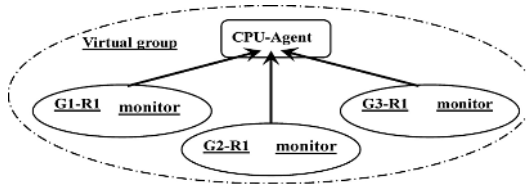


Fig. 2. Virtual Group Based on CPU -Agent

By using the virtual group, we can gain some advantages as follows.

- **Efficient load monitoring:** To the members of the virtual group, they have the same load metric, and we can make use of the agent to avoid the redundant load monitoring. The agent does the actual monitoring and the replicas get the load information from the agent periodically. However, different from the traditional per-replica (host) granularity load balancing middleware, the agent and the replicas may have different periods. The agent may have smaller monitoring period so as to provide more accurate load information and the replicas have bigger periods to bring down the unnecessary overhead.
- **Flexible multiple granularity load monitoring:** The virtual group is based on the similar load metric. If the replicas residing in the same host have different load metric, the load monitoring granularity may be per-replica; if all the

replicas pay attention to the same load metric, the granularity may be per-host. Besides these two conditions, the granularity is per-group (virtual).By this way we can realize flexible multiple-granularity load monitoring.

- **Better scalability:** After new kind of replica is deployed, it will register to the agents that are relevant with its load metric. Although the new replica will affect the load information of the existing replicas, the load information will not invalidate because the agent and the replicas have different monitoring period. The agent will react to the deployment of the new replica and the other replicas will get stable load information later. So the system will get better scalability.

### 2.2 Agent Based Multiple-Granularity Load Balancing Model

In general load balancing systems, we can use CPU utilization, meory utilization, the number of active transactions, the network bandwidth and so on. Most of the load metrics are focusing on the common information of the host, so we can deploy according agents and organizing the replicas into different virtual groups.

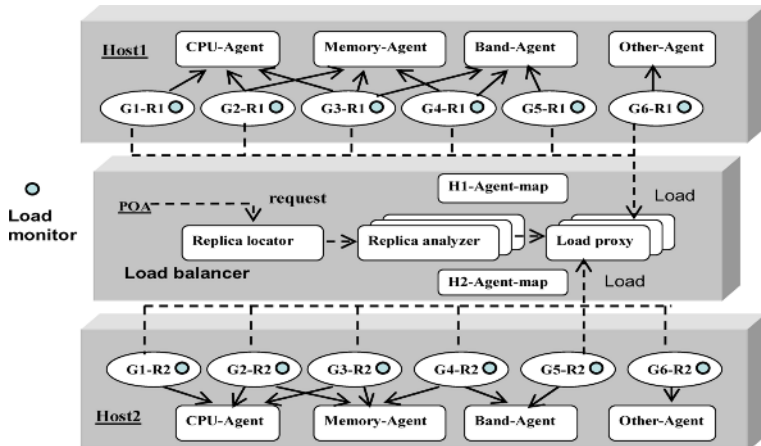


Fig. 3. Agent Based Multiple-Granularity Load-Balancing Middleware Model

In figure 3 is our agent based multiple-granularity load balancing middleware model, we deploy the CPU-agent, memory-agent, band-agent to the hosts and the replicas may get load information from according agents. A replica may connect with one kind of agent or multiple kinds of agents such as G1-R1 (Group1 replica1), G2-R2 (Group1 replica1) and so on. The agents complete the actual load monitoring with smaller period while the replicas get load information from the agents with longer period. At the same time, we can deploy different kinds of agents according to the need of different applications. After deployment, the agents will register with the load balancer and the references of the agents will be written into the Agent-maps. When new replica is deployed, it will get according references of the agents from the load balancer. The function of each component in our model is outline below:

**Replica locator:** This component identifies which replica will receive the client request and it is also in charge of binding the clients to the identified replicas. The replica locator can be implemented using standard CORBA POA mechanisms [8].

**Load analyzer:** This component decides which replica will receive the next request. The replica locator obtains a reference to a replica from the load analyzer and then forwards the request to that replica. The load analyzer allows a load balancing strategy to be selected explicitly at run-time to maintain a simple and flexible design.

**Replica proxy:** Each object managed by load balancer communicates with it via a unique proxy. The load balancer uses these proxies to distinguish different replicas.

**Load balancer:** This component is a mediator that integrates all the components described above. It provides an interface through which load balancing can be administered, without exposing clients to the intricate interactions between the components.

**Agent map:** This component is used by load balancer to store all the references of the agents. When new replica is deployed, according load monitors will connect with the load balancer and get the references if needed.

**Load monitor:** This component monitors load on a given replica as well as according agents, reports replica loads to a load balancer, and responds to load advisories sent by the load balancer. A load monitor can be configured with pull or push policy.

A client obtains an object reference to what appears to be a replica and invokes an operation. In actuality, however, the client transparently invokes the request on the load balancer itself. After the request is received from the client; the load balancer's POA dispatches the request to its servant locator, *i.e.*, the replica locator component. Next, the replica locator queries its load analyzer for an appropriate replica. Depending on the load reporting policy, the load monitor will either report the load to the balancer or the load balancer will query the load monitor for the load. Load monitor gets load information from the replica and the agents periodically. As loads are collected by the load balancer, the load analyzer analyzes the load on the replica and determines whether the replica is overloaded. The replica locator transparently redirects the client to the chosen replica. Requests will be sent directly to the chosen replica until the load balancer detects a high load. The indirection and overhead incurred by per-request load balancing architectures is eliminated since the client communicates with the replica directly.

### 3 Performance Results

Our agent based multiple-granularity middleware StarLB is running on RedHat Linux 9/Pentium4/256/1.7G. All the servers and the clients are running on Windows2000/Pentium4/512M/2G. All the machines are connected through a 100 Mbps ethernet switch.

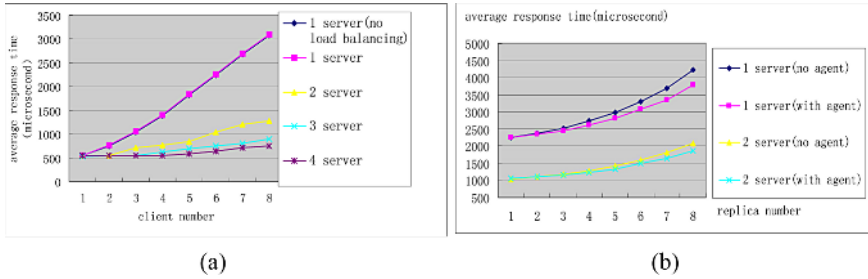


Fig. 4. (a) One Service Replica in One Host (b) Multiple Kinds of Service Replicas in One Host

As depicted in figure 4(a), there is only one replica residing in each host. From broken line 1 and broken line 2 we can see that whether we use load balancing or not there is just a little difference of the average response time and the overhead of load balancer is low. With the increasing of the number of servers, the average response time will decrease and the overall throughput will increase.

Furthermore, thinking about the conditions that there are multiple kinds of service replicas residing in one host. Supposing there are six independent clients requesting for service and the number of the replicas in each server increased by one every time. From the broken lines 1 to broken lines 4 in figure 4(b) we can see whether there is one server or two servers the average response time will increase after more and more replicas have been deployed because of the overhead of more monitoring and more client requests. At the same time, we can see that by using agents the average response time will relatively lower than that having no affection of the agents which means that the overhead of monitoring will be decreased effectively. All the results are depicted in figure 4(b).

## 4 Conclusions

To alleviate overload by the volume of client requests, load balancing mechanisms can be used to distribute system load across object replicas residing on multiple servers. Load can be balanced at several levels, including the network, OS, and middleware. But to many distributed applications, Network-based and OS-based load balancing architectures may suffer from several limitations, and middleware-based load balancing architectures particularly adaptive load balancing middleware have been applied widely. However, most of the current load-balancing middleware adopt the per-object load monitoring granularity, so when multiple service groups exist in the same host problems will arise. In this paper, we present and implement an agent based multiple-granularity load balancing middleware model by using design patterns and the key design challenges including adjustable load monitoring granularity, customizable load balancing strategy, load feeding back and adaptive control. As a result, developers can concentrate on their core application behavior, rather than wrestling with complex infrastructure mechanisms needed to make their application distributed, scalable, and dependable.

## References

- [1] Rajkumar Buyya. "High Performance Cluster Computing Architecture and Systems", ISBN7.5053-6770-6.2001.
- [2] F.Douglis and J.Ousterhoot, "Process Migration in the Sprite Operation System". Proceedings of the 70' International Conference on Distributed Systems, (Berlin, West Germany), pp.18-25, IEEE, Sept. 1987.
- [3] S.M.Baker and B.Moon. "Distributed cooperative web servers", Proceedings of the 8th International WWW Conference, Toronto, Canada. May 11-14, 1999.
- [4] O.Othman, C. O'Ryan, and D. C. Schmidt, "The Design of an Adaptive CORBA Load Balancing Service, " IEEE Distributed Systems Online, vol. 2, Apr. 2001.
- [5] Othman O, Schmidt DC. Issues in the design of adaptive middleware load balancing. In: ACM SIGPLAN, ed. Proceedings of the ACM SIGPLAN workshop on Languages, Compilers and Tools for Embedded Systems. New York: ACM Press, 2001. 205~213.
- [6] Othman O, O'Ryan C, Schmidt DC. Strategies for CORBA middleware-based load balancing. IEEE Distributed Systems Online, 2001, 2(3). <http://www.computer.org/dsonline>.
- [7] IONA Technologies, "Orbix 2000." [www.iona-portal.com/suite/orbix2000.htm](http://www.iona-portal.com/suite/orbix2000.htm).
- [8] M. Henning and S. Vinoski, Advanced CORBA Programming With C++. Addison-Wesley Longman, 1999.
- [9] Object Management Group, The Common Object Request Broker: Architecture and Specification, 3.0 ed., June 2002.
- [10] Gamma E, Helm R, Johnson R, Vlissides J. Design Patterns: Elements of Reusable Object-Oriented Software. Reading: Addison-Wesley, 2002. 223~325.

# Multiagent Model for Grid Computing\*

Qingkui Chen<sup>1</sup> and Lichun Na<sup>2</sup>

<sup>1</sup> School of Computer Engineering, University of Shanghai for Science and Technology,  
Shanghai 200093, China  
chenqingkui@tom.com, chenqingk@sohu.com

<sup>2</sup> Dept. of Information Science, Shanghai LIXIN University of Commerce,  
Shanghai 201620, China  
Nalc@sohu.com

**Abstract.** For supporting the grid computing in dynamic network environment, a multiagent model is proposed in this paper. A series of formal definitions, such as the architecture of the model, the dynamic network environment (DNE), the manage agent system, the independent computing agents (ICA) which support the traditional computing base on migration, the cooperation computing team (CCT) which supports the data parallel computing, and the relations among them are given. The dynamic learning method and the fuzzy partition technique for logical computer cluster, on which the CCT runs, are studied. The computing process is described. The experiment results show that this model resolves effectively the problems of optimization use of resources in DNE. It can be fit for grid computing.

## 1 Introduction

With the rapid development of the information techniques and their popular applications, the demand for the high performance processing devices is becoming more and more vehement. Nowadays, the numbers of Intranet composed of many computer clusters are quickly increasing, and a great deal of cheap personal computers are distributed everywhere, but the using rate of their resources is very low [1, 2]. The grid [3] techniques can become the main approach to use effectively these resources. Mining and adopting these idle resources, we can get a lot of large-scale high performance computation, storage and communication resources which are not special. However, the heterogeneous, dynamic and unstable characteristic of these resources brings the huge obstacle for us. The multiagents [4, 5] have already become the feasible solutions for grid computing. There are many successful examples [6, 7] of applications which are in conjunction with the automatic multiagent system. However, the multiagent model for grid computing is still not much in dynamic network environment (*DNE*) that composed of many computer-clusters connected by Intranet.

This paper introduced a Multiagent Model for Grid Computing (MMGC) in *DNE*. Building an open grid computing environment, using of the idle computational

---

\* This paper is supported by the National Nature Science Foundation of China (No.60573108) and the Nature Science Foundation of Shanghai of China (No.04ZR14100).

resources of *DNE*, and adopting the fuzzy theory [8] and the learning methods [9,10], we designed the MMGC model that can support the Data Parallel Computing (DPC) and the traditional computing (TC) based on migration mechanism. The experimental results show that MMGC can increase the percentage of the using resources in *DNE* and it can improve response time of computation-intensive tasks.

## 2 Architecture of MMGC

MMGC includes two parts: one is *DNE* that is the physical computing devices for grid computing, and *DNE* is composed of many computer clusters connected through LAN and Intranet, and all the computers is not special; other is the agent system, and we call it as *GCG* (Global Computing Group). *GCG* is composed of a *manage agent system (MAS)* and a lots of *computing agents*. The architecture is presented in figure1. MMGC supports two types of computing tasks: the one is the traditional computing (TC); the other is the data parallel computing (DPC) [11] based on the logical computer cluster.

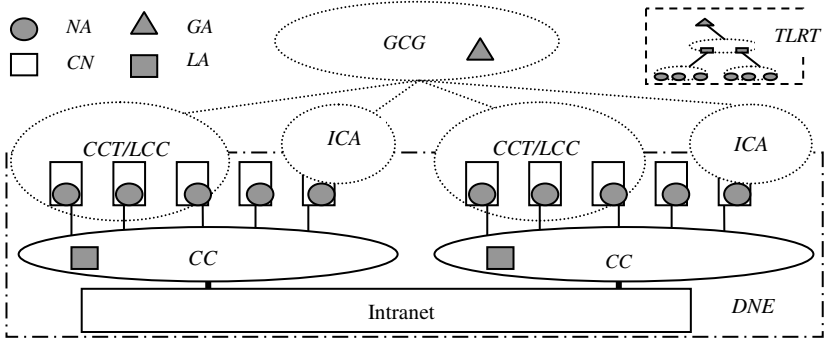


Fig. 1. The architecture of MMGC

## 3 DNE

The main components of *DNE* are described as follows:

- (1) **Computing node (CN)** is defined as  $CN(id, CT, AS)$ , where *id* is the identifier of *CN*; *CT* is the type of computing node; *AS* is the set of agents running on *CN*;
- (2) **Computer cluster (CC)** is defined as  $CC(Master, CS)$ , where *Master* is the main computer of *CC*;  $CS = \{CN_1, CN_2 \dots CN_p\}$  is the set of all computing nodes which *CC* includes.
- (3) **Logical computer cluster (LCC)** is defined as  $LCC(id, LCS, B, CC)$ , where *id* is the identifier of *LCC*; *LCS* is the set of computing nodes of *LCC*; *CC* is the computer cluster which includes *LCC*. *B* is the network bandwidth of *LCC*.



So, the **dynamic network environment (DNE)** can be defined as  $DNE (Master, CCS, N, R)$ , where Master is the main computer of DNE; CCS is the set of all computer clusters which DNE includes; N is its network set; R is the connection rules.

Because of the difference resources (which CC, CN, and the network provided) of DNE, their types must be considered. These types are described as follows:

- (1) **Computing node type (CNT)** is defined as  $CNT (cpu, mem, disk, net)$ , where *cpu* is the processor power of CN; *mem* is the storage power of CN; *disk* is the I/O power of CN; *net* is the communication power of CN. According to the real conditions of each CN in DNE, the CN types can be formed into the type set  $CTS = \{CT_1, CT_2, \dots, CT_{ct}\}$ , and the CTS is called as the **set of computing node type**;
- (2) **Network type (NT)** is defined as  $NT (B, PRT)$ , where *B* is the network bandwidth of CC; *PRT* is the network protocols of CC. By the real condition of networks, the network types can be formed into the type set  $NTS = \{NT_1, NT_2, \dots, NT_{nt}\}$ .

## 4 Manage Agents System (MAS)

The main functions of MAS are the computation resource management, the DNE monitoring and the task scheduler. MAS include three parts: The first part is the global control agent (GA) for managing MMGC. The second part is the agent for managing one computer cluster, and it is called as the local agent (LA). The third part is the agents for managing the computing nodes, and they are called as the node Agents (NA), and each computing node has a NA. MAS' structure is the tree-level-ring-tree (TLRT) and it is shown in Figure 1.

The main functions of GA are as follows: (1) Control and manage DNE; (2) Receive and dispatch the computing tasks; (3) Construct and remove CCT for DPC.

The main functions of LA are as follows: (1) Control all the computing nodes in its cluster; (2) Calculate the idle resources of cluster, and report them to GA; (3) Monitor the states all computing nodes and node agents; (4) construct the LCC for DPC task;

The main functions of NA are as follows: (1) Control the computing node join or disjoin the DNE in dynamic; (2) Calculate the idle resources, and report them to LA; (3) Monitor the states and events in CN, and make the response adjustments; (4) Control the computing agents to complete the computing task.

## 5 Computing Agents

For supporting the DPC and TC, we introduce some computing agent conceptions:

- (1) **Computing agent (CA)** is defined as  $CA (id, PRG, BDI, KS, CE)$ , where *id* is the identifier of CA; *PRG* is the executable program set of CA; *BDI* is the description of its BDI; *KS* is its knowledge set; *CE* is its configuration environment. CA is the basic element to execute computation task;
- (2) If a CA could complete independently the task, we call it as the **independent computing agent (ICA)**;
- (3) If a CA couldn't complete independently the task, and it must cooperate with others, we call it as the **cooperative computing agent (CCA)**;

(4) **Cooperation computing team (CCT)** is defined as  $CCT(id, Am, CAS, BDI, CKS, CCE, LCC)$ , where  $id$  is the identifier of  $CCT$ ;  $Am$  its main control agent;  $CAS$  is the set of all cooperative computing agents ( $CCA$ ) which  $CCT$  includes;  $BDI$  is the description of its  $BDI$ ;  $CKS$  is its knowledge set.  $CCE$  is its configuration environment;  $LCC$  is a logical computer cluster, on which  $CCT$  runs.

So, the **global computing group (GCG)** is defined as  $GCG(id, MAS, ICAS, CCTS, GKS, GCE)$ , where  $MAS$  is the manage agent system of  $GCG$ ; where  $id$  is the identifier of  $GCG$ ;  $ICAS$  is the set of  $ICA$  which  $GCG$  includes;  $CCTS$  is the set of  $CCT$  which  $GCG$  includes;  $GKS$  is its knowledge set.  $GCE$  is its configuration environment. The relations between  $DNE$  and  $GCG$  are presented in figure 1.

A **Task (TSK)** is defined as  $TSK(id, RDV, DAT, AS, St)$ , where  $id$  is the identifier of  $TSK$ ;  $RDV(cpu, mem, disk, net)$  is its **resource demand vector**;  $DAT$  is the data set;  $AS$  is the set of agents to calculate  $TSK$ .  $St$  is its state.  $TST$  has six states, and these states are “Committed”, “ready”, “Suspended”, “Migrating”, “Running” and “Completed”.

## 6 Partition Methods for Logical Computer Cluster

Suppose that  $CC$  is a computer cluster and  $LA$  is the local manage agent on  $CC$ . When  $LA$  receives the information from all  $NAs$  which  $CC$  includes, it will construct the logical computer cluster for DPC task. Because of the difference of resources provided by computer nodes, we must classify the computer nodes in order to form the different power  $LCC$ . The classification is based on their power of cpu, memory, disk, and net adapter. In order to solve this problem, we discrete the power data and we adopt the fuzzy relation theory. The  $LCC$  partition process is described as follows:

- (1)  $MR = (r_{ij}) (1 \leq i \leq n, 1 \leq j \leq 4)$  is the resource matrix in  $CC$ , where  $n$  is the numbers of computing nodes in  $CC$ ;  $r_{i1}$  is the computing power of  $CN_i$ ;  $r_{i2}$  is the storage power of  $CN_i$ ;  $r_{i3}$  is the I/O power of  $CN_i$ ;  $r_{i4}$  is the communication power of  $CN_i$ ;  $TSK$  is a DPC task;
- (2) Construct the fuzzy matrix  $FM = (f_{ij}) (1 \leq i \leq n, 1 \leq j \leq n)$ , where  
 $f_{ii} = 1$ ;  
 $f_{ij} = 1 - \beta (w_1 |r_{i1} - r_{j1}| + w_2 |r_{i2} - r_{j2}| + w_3 |r_{i3} - r_{j3}| + w_4 |r_{i4} - r_{j4}|)$ , when  $i \neq j$ , and  
 $0 < \beta < 1$ , and  $w_1 + w_2 + w_3 + w_4 = 1, w_k > 0 (1 \leq k \leq 4)$ ;
- (3) build the fuzzy equivalence matrix:  
 Repeat do { $FT = FM \odot FM$ ; If  $FT = FM$  then goto (4);  $FM = FT$ ;}  
 $// \odot$  is the operation theorem to take the maximum and minimum
- (4) Calculate the  $c$ -cut matrix  $FM_c$  by the free network bandwidth in  $CC$ ;
- (5) Divide the  $CNs$  into several equivalence class  $LCC_1 \cup LCC_2 \cup \dots \cup LCC_e$  by  $FM_c$ ;
- (6) Choose a  $LCC$  for  $TSK$  according to its resource demand vector.

## 7 Description of Agent Learning Model

The agent rules are described as follows:

- (1) **Basic rule ( $br$ )** is defined as  $br(id, rul, MRS)$ , where  $id$  is its identifier;  $rul$  is the description of  $br$ ;  $MRS$  is the meta-rules set for revising  $br$ ; The **basic rule set ( $BRS$ )** is the set of all basic rules that  $GCG$  includes;
- (2) **Dynamic rule ( $dr$ )** is defined as  $dr(ct, nt, br, rul, w, sta, life)$ , where  $ct \in CTS$ ,  $nt \in NTS$ ,  $br \in BRS$ ;  $rul$  is the formalization description of  $dr$ ;  $w$  is the its weight value; and  $sta$  is its state, and  $sta \in \{“Naive”, “Trainable”, “Stable”\}$ ; “Naive” denotes that the  $dr$  is a new rule; “Trainable” denotes that the  $dr$  is revising rule; “Stable” denotes that the  $dr$  is a mature rule;  $life$  is the its life value;
- (3) If  $dr$  is a dynamic rule and  $dr.w > MaxWeight$ , which  $MaxWeight$  is a constant in  $GCG$ , we call  $dr$  as the **static rule ( $sr$ )**, its state is “Static”;
- (4) If  $dr$  is a dynamic rule and  $dr.w < MinWeight$ , which  $MinWeight$  is a constant in  $GCG$ , we call  $dr$  as **castoff rule ( $cr$ )**. Its state is “Castoff”.

The state graph of rules is presented in figure 2.

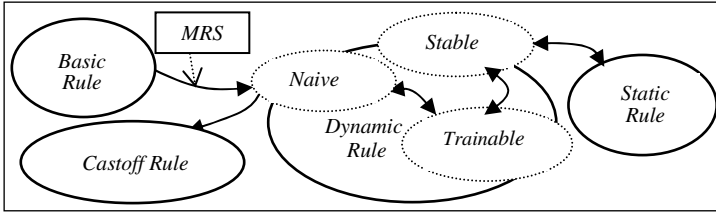


Fig. 2. The state graph of rules

The dynamic knowledge is the set of all the dynamic rules in  $GCG$ . The static knowledge is the set of all static rules. The basic knowledge can be formed by passive learning. For fitting the variety of computing resources, the dynamic rules must be generated at the start of the computing and the dynamic rules must be revised during the  $TSK$  computing. The revising mechanism can adopt the reinforcement learning, and the use rate of resources for  $TSK$  can be as the reinforcement factors.

Suppose that  $Y_1$  is the **castoff threshold**, and  $Y_2$  is the **mature threshold**;  $Q(urt)$  is the **reinforcement function**, and  $Q(urt) > 0$ , and  $urt$  is the use rate of resources;  $MaxWeight$  is the maximum of the rule weight, and  $MinWeight$  is the minimum of the rule weight, and let  $MinWeight < Y_1 < Y_2 < MaxWeight$ ;  $MaxLife$  is the maximum of life value. The revising process is as follows:

- (1) Suppose that a computing agent  $CA$  adopted a dynamic rule  $dr$  of  $CA.KS$ ;
- (2)  $dr.life + +$ ; //increase the value of life
- (3) wait for the  $urt$  from  $MAS$ ;
- (4) If  $urt > 0$  then  $dr.w = dr.w + Q(urt)$ ; //increase weight  
If  $urt < 0$  then  $dr.w = dr.w - Q(urt)$ ; //decrease weight
- (5) If  $dr.w > MaxWeight$  then  $dr.w = MaxWeight$ ;  
If  $dr.w < MinWeight$  then  $dr.w = MinWeight$ ;

- (6) If  $dr.w < Y_1$  and  $dr.life > MaxLife$  then  $dr.sta = "Castoff"$ ; //Castoff rule
- If  $Y_2 < dr.w < MaxWeight$  then  $dr.sta = "Stable"$ ; //Stable rule
- If  $dr.w \geq MaxWeight$  then  $dr.sta = "Static"$ ; //Static rule
- If  $Y_1 < dr.w < Y_2$  then  $dr.sta = "Trainable"$ ; //Trainable rule
- If  $MinWeight < dr.w < Y_1$  then  $dr.sta = "Naive"$ . //Naive rule

## 8 Process of MMGC

The MMGC computing process is as follows:

- (1) The user agent commits the tasks to *GCG*, and their states is "*Committed*";
- (2) *GCG* gets a task *TSK* that state is "*Ready*";
- (3) *GCG* does the initialization works for *TSK*;
- (4) *CGG* allots a computing device *PE* (that is a *CN* or a *LCC*) for *TSK*; if *TSK* is *DPC*, *GCG* get a *LCC* (That is constructed by *LA* through fuzzy partition method) for *TSK*;
- (5) *GCG* constructed computing Agents (*ICA* or *CCT*) for *TSK*;
- (6) The computing agents (*ICA* or *CCT*) for *TSK* construct the dynamic knowledge;
- (7) All computing Agents calculate the *TSK* in cooperative, and they start the functions to revise the dynamic knowledge;
- (8) If the *TSK* must be migrated, *GCG* start the migration process and do the step (6) ;
- (9) If *TSK* be finished , *GCG* receive the *TSK* from *NA* of *PE* and save the new knowledge into its knowledge base .

## 9 Experiments

In order to test MMGC, We built a *DNE* that is composed of 24 computers and 2 computer-clusters. All the computers are classified into 6 types .The computing tasks are the matrix operations, the linear programming and the JOIN operation. We programmed the TC edition and DPC edition. The Matrix operation *RDV* is (0.3, 0.3, 0.2, 0.2), and the linear programming *RDV* is (0.5, 0.3, 0.1, 0.1), and the JOIN *RDV* is (0.2, 0.3, 0.25, 0.25). The initialization basic rules include 24 rules for *CA* and 7 rules for *CCT*, and the parameters are as follows:  $MaxLife=43200(s)$ ;  $Y_1=15; Y_2=80; Y_3=0.3$ ;  $MaxWeight=100$ ;  $MinWeight=0$ ;  $r_{i1} \in \{1,2,3,4,5,6\}$ ;  $r_{i2} \in \{1,2,3,4,5\}$ ;  $r_{i3} \in \{1,2,3\}$ ;  $r_{i4} \in \{1,2,3\}$ .The tests include seven times, and each time has 12 hours. The tests adopt a random function to choose some tasks in each time. The results are shown in Figure 3.

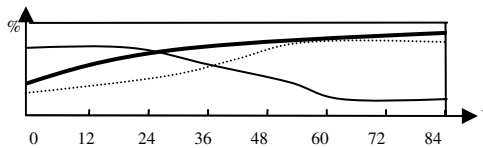


Fig. 3. The results of tests

The variety of the average use rates of *DNE* resources along with the computing process is presented by the thick solid line, and it shows that MMGC can raise the use rates of *DNE* resources consumedly. The variety of the numbers of new dynamic rules, which are generated during the learning process, had been tested. The solid line

is the distribution of the dynamic rules that their state is always “*Naive*” during their life period. The dotted line is the distribution of the “*Trainable*” dynamic rules that their state had become the “*Stable*” during their life period. This test results show that the learning efficiency of this model increases gradually along with computing process. The effective scales of *LCC* for these three types of DPC had been tested. The scale for Matrix operation is about 8, for Liner programming is about 10, and for JOIN operation is about 5.

## 10 Conclusions

Because of the heterogeneous resources, the different power of computers, and the migration of computing task, the effective use of resources is very difficult for the grid computing in the dynamic network environment. Through the techniques of *ICA*, *CCT*, the Dynamic learning and the logical computer cluster partition based on fuzzy theory, MMGC can solve these problems. And MMGC can raise the use rate of resources in *DNE*. It can fit for the grid computing in *DNE*.

## References

1. E. P. Markatos and G. Dramitios: Implementation of Reliable Remote Memory Pager. In proceedings of the 1996 Usenix technical Conference, (1996) 177-190
2. Anurag Acharya and Sanjeev Setia: Using Idle Memory for Data-Intensive Computations. In proceedings of the 1998 ACM SIGMETRICS Conference on Measurement and Modeling of Computer Systems, (1998)278-279
3. I. Foster, C. Kesselman: The Grid: Blueprint for Future Computing Infrastructure. San Francisco, USA: Morgan Kaufmann Publishers, (1999)
4. E. Osawa: A Scheme for Agent Collaboration in Open MultiAgent Environment. Proceeding of IJCAI'93, (1993)352-358
5. Wooldridge, M.: An Introduction to Multivalent System, John Wiley & Sons (Chichester, England). ISBN 0 47149691X, (2002)
6. O.F. Rana and D.W. Walker: The Agent Grid: Agent-based resource integration in PSEs, In proceedings of the 16th IMACS World congress on Scientific Computing, Applied mathematics and Simulation, Lausanne, Switzerland, (2000 )
7. J.O. Kephart, Chess, D.M: The Vision of Autonomic Computing, IEEE Computer, (2003) 41-50
8. L. A. Zadeh: The concept of a linguistic variable and its application to approximate reasoning, American Elsevier Publishing Company, Inc, (1975)
9. L. P. Kaelbling, M. L. Littman, and Moore: Reinforcement learning: A survey. Journal of Artificial Intelligence Research, (1996), 4(2):237-285
10. F. Zambonelli, N. R. Jennings, M. Wooldridge: Organizational rules as abstractions for the analysis and design of multi-agent systems. International Journal of Software Engineering and Knowledge Engineering, (2001), 11(3):303-328
11. Rajkumar Buyya: High performance Cluster Computing Architectures and Systems, Prentice Hall, (1999)

# Using Two Main Arguments in Agent Negotiation

Jinghua Wu<sup>1</sup>, Guorui Jiang<sup>1</sup>, and Tiyun Huang<sup>1,2</sup>

<sup>1</sup> Economics and Management School, Beijing University of Technology, 100 Pingleyuan, Chaoyang District, Beijing, 10022 P.R. China  
uwhua@163.com, jianggr@bjut.edu.cn

<sup>2</sup> Management School, Harbin Institute of Technology, 92 West Da-Zhi Street, Harbin, Heilongjiang, 150001 P.R. China  
tyhuang@hit.edu.cn

**Abstract.** During agent negotiation, argumentation-based negotiation of agent has been widely studied for it can make agent who received the argument change its goals or preferences accordingly. Being the two main arguments, threat and reward can even make the negotiators reduce their behavior space to find a well compromise quickly in the end, which can make them accomplish their cooperation on the base of getting the most profit of each of them. This paper presents a type of formal models of threat and reward first, and then present a new way of how to calculate the negotiation strengths of them through simulated calculations based on the models to make the negotiators threatened or rewarded make a right choice and accomplish their cooperation well.

**Keywords:** Threat, Reward, Agent, Negotiation.

## 1 Introduction

Argumentation-based negotiation of agent [1, 2, 3] can make the negotiators exchange additional information that has not been expressed in the proposal, so it has been widely studied. Being the two main arguments, threat and reward can even make negotiators reduce their behavior space to find a well compromise quickly in the end, which can make them accomplish their cooperation on the base of getting the most profit of each of them. And it can also make their final decision-making [4] well.

In this area, [5] is classical, which has integrated four main parameters that can represent the mental state of agent as belief, desire, intention and goal with the definition of threat and reward, and then advanced a kind of logical model of them. But at first, the model hasn't dealt with the evaluation of the importance of its opponent such as knowledge; secondly, the parameters in this model just limited to mental and time, and it hasn't dealt with the evaluation of the negotiation content such as threat and reward itself; finally, the parameters are so many and complex that make the model difficult to be understood.

After that, many studies about the generation [6], evaluation [7], selection [8] of threat and reward has more thought much of its calculation only in simple calculation of mathematics, which has made the function of agent in negotiation limited to simple exchanges of proposals and counter-proposals.

Now in [9], Amgoud has advanced a kind of formal model of threat and reward based on [5]. In their models, they have mainly made level with the information of agent about its context, then evaluated it by the maximum and minimum of corresponding weight after it has been put forward, which remedied some shortages of [5] to a certain extent. But the level has been made so simplest and it couldn't achieve the goal of quantification of threat and reward, which has little effect on evaluating them; on the other side, they haven't set up relative models of their evaluation and analyze their negotiation strength.

Besides this, [10, 11, 12] has studied threat and reward solely. [10] is only about passive threat, which is an aspect of threat. [11, 12] has presented a way of evaluation about reward based on the learning of agent, whether their models can suit other forms of threat or reward has not been verified yet.

Based on this background, this paper aims at providing a type of formal models of threat and reward and present a new way of how to calculate the negotiation strengths of them. After that, some examples are proposed to explain the former, and some simulated calculation and analysis are proposed to verify the validity of the latter.

This paper is organized as follows: Section 2 presents a type of formal models of threat and reward after definite and classify them, then gives some examples to explain them; Section 3 presents a new way of how to calculate the negotiation strengths of threat and reward based on the models, then makes some simulated calculations and analysis to verify their validity; Section 4 is the conclusions.

## 2 Models of Threat and Reward

### 2.1 Definitions and Classifications

During the course of negotiation, when it is being brought into a stalemate for one's items of negotiation are conflicted with the goal or intention of another one, this agent may use it as a force (or an incitement) to persuade its opponent to do (or not to do) the business with it, which goal is to accomplish the cooperation between them, and at the same time get the most profit of both of them.

Usually a threat (reward) can be classified as two main kinds as follows [5]:

1. You should do this otherwise I will do that, which is bad (good) for you;
2. You should not do this otherwise I will do that, which is bad (good) for you.

### 2.2 Models

For the negotiation side of threatening (or rewarding)  $A$ ,  $Threat(A \Rightarrow B)$  ( $Reward(A \Rightarrow B)$ ) can be formally modeled as a tuple  $\langle A, B, K, G, G' \rangle$ , and that:

1.  $A$  is the negotiation side of threatening (rewarding),  $B$  is the other negotiation side of threatened (rewarded);
2.  $K$  is the knowledge relative to this threat (reward) of  $A$ ,  $K \in K_A$  ;
3.  $G$  is the goal  $A$  want to achieve by proposing this threat (reward),  $G \in G_A$  ;

4.  $G'$  is the goal  $A$  think  $B$  want to achieve after  $B$  has accepted this threat (reward),  $G' \in G'_A$ ;
5. During the course of negotiation, for the threat, there is always  $K \wedge \neg G \bullet \neg G'$  ; for the reward, there is always  $K \wedge G \bullet G'$  .  $K \wedge G$  is consistent.

For the other negotiation side of threatened (rewarded)  $B$  ,  $Threat(A \Rightarrow B)$  ( $Re\ reward(A \Rightarrow B)$ ) can also be formally modeled with the same rule, but  $K \in K_B$ 、  $G \in G_B$ 、  $G' \in G'_B$  .

### 2.3 Examples

**Example 1 (Threat).** During the course of agent negotiation, the order has been given by a buyer agent  $A$  is quite different with expectation of a seller agent  $B$  , the former request the latter accept it ( $AcptOrd$ ) . But for its most profit,  $B$  may reject it, which may bring the negotiation into a stalemate. In this status, to guarantee the most profit of both of them and the negotiation to be continued successfully,  $A$  may propose some threats such as stopping every deal ( $StopDeal$ ) with  $B$  to threaten  $B$  to accept, and finally accomplish the business to achieve the cooperation between them.

So for  $A$  , this threat can be formally modeled as:

$$Threat(A \Rightarrow B) = \langle A, B, \{ \neg AcptOrd \rightarrow StopDeal \}, AcptOrd, \neg StopDeal \rangle$$

1.  $K = \langle \neg AcptOrd \rightarrow StopDeal \rangle, K \in K_A$  ;
2.  $G = \langle AcptOrd \rangle, G \in G_A$  ;
3.  $G' = \langle \neg StopDeal \rangle, G' \in G'_A$  .

For  $B$  , the model of this threat can be set up in the same way.

**Example 2 (Reward).** In the stalemate of negotiation in example1,  $A$  may propose some rewards such as buy some other products ( $BuyOth$ ) from  $B$  to incite  $B$  to accept, and finally accomplish the business to achieve the cooperation between them.

So for  $A$  , this reward can be formally modeled as:

$$Re\ ward(A \Rightarrow B) = \langle A, B, \{ AcptOrd \rightarrow BuyOth \}, AcptOrd, BuyOth \rangle$$

1.  $K = \langle AcptOrd \rightarrow BuyOth \rangle, K \in K_A$  ;
2.  $G = \langle AcptOrd \rangle, G \in G_A$  ;
3.  $G' = \langle BuyOth \rangle, G' \in G'_A$  .

For  $B$  , the model of this reward can be set up in the same way.



### 3 Calculation of Negotiation Strengths of Threat and Reward

The formal models of threat (reward) upwards include just two negotiators, and the negotiation side of threatened (rewarded) is usually willing to accept it to accomplish the cooperation of them and get the most profit of both of them. But during the course of negotiation, the numbers of negotiators and threats (rewards) can all be many and many. At this time, also to guarantee the most profit of both of them and the negotiation to be continued successfully, the negotiator which has been encountered with several threats (rewards) should make quantification with each threat (reward) according to its relative knowledge, and then calculate its value to compare the negotiation strengths of each threat (reward).

#### 3.1 Calculation of Negotiation Strengths of Threat (Reward)

Let  $\Sigma$  be a collection of all the threats (rewards) that  $X$  have been encountered from  $A$ 、 $B$ 、 $C$ 、 $D$  at the same time, such that:

$$\begin{aligned} Threat(A \Rightarrow X) \in \Sigma, \quad Threat(B \Rightarrow X) \in \Sigma, \\ Reward(C \Rightarrow X) \in \Sigma, \quad Reward(D \Rightarrow X) \in \Sigma \end{aligned}$$

The calculations of negotiation strengths of threat from  $A$  and reward from  $C$  are:

$$VI_{StrTh}(A \Rightarrow X) = \omega_{\Phi} \times E(\Phi_A) + \omega_G \times E(G_A) + \omega_{G'} \times E(G'_A) \quad (1)$$

$$VI_{StrRw}(C \Rightarrow X) = \omega_{\Phi} \times E(\Phi_C) + \omega_G \times E(G_C) + \omega_{G'} \times E(G'_C) \quad (2)$$

1.  $\Phi_A$  ( $\Phi_C$ ) is the negotiation side of threatening (rewarding);
2.  $G_A$  ( $G_C$ ) is the goal the negotiation side of threatening (rewarding) wants;
3.  $G'_A$  ( $G'_C$ ) is the goal the negotiation side of threatened (rewarded) thinks itself wants after it has accepted this threat (reward);
4.  $E(\Phi_A)$ 、 $E(G_A)$ 、 $E(G'_A)$  are evaluation values of  $\Phi_A$ 、 $G_A$ 、 $G'_A$  calculated by  $X$ , and so do  $E(\Phi_C)$ 、 $E(G_C)$ 、 $E(G'_C)$ ;
5.  $\omega_{\Phi}$ 、 $\omega_G$ 、 $\omega_{G'}$  are weight values of  $E(\Phi_A)$ 、 $E(G_A)$ 、 $E(G'_A)$  of  $X$  according to its relative knowledge, and so does  $C$ .

The calculations of negotiation strengths of  $Threat(B \Rightarrow X)$ 、 $Reward(D \Rightarrow X)$  can also be calculated in the same way.

#### 3.2 Simulated Calculation and Analysis

The evaluation of  $E(\Phi_A)$ 、 $E(\Phi_B)$ 、 $E(\Phi_C)$ 、 $E(\Phi_D)$  can be decomposed just as credit degree、the number of successful deals、satisfaction degrees; The evaluation of  $E(G_A)$ 、 $E(G_B)$ 、 $E(G_C)$ 、 $E(G_D)$  can be decomposed just as price quantity

delivery they are negotiating, and so does  $E(G'_A)$ ,  $E(G'_B)$ ,  $E(G'_C)$ ,  $E(G'_D)$ . Corresponding data and weight values can be given as table 1, 2, 3, and the negotiation strength of every threat (reward) can be calculated as table 4.

**Table 1.** Corresponding data and weight values of  $E(\Phi_A)$ ,  $E(\Phi_B)$ ,  $E(\Phi_C)$ ,  $E(\Phi_D)$

	<i>AgentA</i>	<i>AgentB</i>	<i>AgentC</i>	<i>AgentD</i>	Weight
Credit	8	6	5	9	0.5
Deals	8	7	8	6	0.3
Satisfaction	6	10	7	5	0.2

**Table 2.** Corresponding data and weight values of  $E(G_A)$ ,  $E(G_B)$ ,  $E(G_C)$ ,  $E(G_D)$

	<i>AgentA</i>	<i>AgentB</i>	<i>AgentC</i>	<i>AgentD</i>	Weight
Price	5	6	4	10	0.4
Quantity	8	5	8	5	0.4
Delivery	7	9	7	6	0.2

**Table 3.** Corresponding data and weight values of  $E(G'_A)$ ,  $E(G'_B)$ ,  $E(G'_C)$ ,  $E(G'_D)$

	<i>AgentA</i>	<i>AgentB</i>	<i>AgentC</i>	<i>AgentD</i>	Weight
Price	9	10	7	8	0.4
Quantity	8	5	6	9	0.3
Delivery	6	8	5	5	0.3

**Table 4.** Calculations of negotiation strength of every threat (reward)

	Evaluation values			Weight values			Negotiation strength
	$E(\Phi)$	$E(G)$	$E(G')$	$\omega_\Phi$	$\omega_G$	$\omega_{G'}$	
<i>AgentA</i>	7.6	6.6	7.8	0.3	0.5	0.2	7.14
<i>AgentB</i>	7.1	6.2	7.9				6.81
<i>AgentC</i>	6.3	6.2	6.1				6.21
<i>AgentD</i>	7.3	7.2	7.4				7.27

So for  $X$ , the negotiation strength of every threat (reward) can be ordered as:

$$VlStrRe(D \Rightarrow X) > VlStrTh(A \Rightarrow X) > VlStrTh(B \Rightarrow X) > VlStrRw(C \Rightarrow X) \quad (3)$$

Finally with this result,  $X$  should deal with  $D$  first, and then deal with other agents by the order of  $A$ ,  $B$ ,  $C$ . And at the same time it can also guarantee the most profit of them and accomplish their cooperation.

## 4 Conclusions

Compared with traditional ways of agent negotiation such as proposal and counter-proposal, threat and reward in argumentation-based negotiation of agent can also make negotiators exchange additional knowledge that can simulate human beings besides such simple knowledge, which can reduce their behavior space of negotiation and make the result willingly accepted by each one of them, and finally accomplish their cooperation on the base of getting the most profit of them. How to effectively model and evaluate with some other kinds of argumentations in agent negotiation such as appeal, rebut, undercut, and etc., can be perspective extensions.

## References

1. Raiffa H. The art and science of negotiation [M]. Harvard University Press, Cambridge (1982)
2. Ramchurn, S.D., Jennings, N.R., Sierra, C.: Persuasive negotiation for autonomous agents: A rhetorical approach. In IJCAIWorkshop on Computational Models of Natural Argument, Acapulco, Mexico (2003) 9–18
3. Iyad Rahwan, Sarvapali D. Ramchurn, Nicholas R. Jennings, Peter Mcburney, Simon Imon Parsons and Liz Sonenberg. Argumentation-based negotiation. The Knowledge Engineering Review. Cambridge University Press 18 (2004) 343–375
4. Tiyun Huang. Intelligence Decision Supporting System[ M ]. Beijing: Electronics Industry Press (2001)
5. Kraus, S., Sycara, K., and Evenchik, A. Reaching agreements through argumentation: a logical model and implementation. Journal of Artificial Intelligence 104 (1998)
6. Rahwan, I., Sonenberg, L., Dignum, F.: Towards interest-based negotiation. In Proceedings of the 2nd International Joint Conference on Autonomas Agents and Multi-Agent Systems (AAMAS'03), Melbourne, Australia (2003) 773–780
7. L.Amgoud and N. Maudet: Strategical considerations for argumentative agents (preliminary report). In Proceedings of the 9th International Workshop on Non-Monotonic Reasoning (NMR'02): Special session on Argument, dialogue, decision, Toulouse, France (2002) 399–407
8. Parsons, S., Sierra, C., Jennings, N.R.: Agents that reason and negotiate by arguing. Journal of Logic and Computation 8 (1998) 261–292
9. L.Amgoud and H.Prad. Formal handling of threats and rewards in a negotiation dialogue. In Proceedings of the 4th International joint Conference on Autonomous Agents and Multi-Agent Systems, AAMAS'2005, Utrecht,. Franck Dignum, Michael Wooldridge, Sven Koenig, Sarit Kraus (Eds.), ACM Press (2005) 529-536
10. Yair B.Weinberger and Jeffrey S. Rosenschein. Passive Threats among Agents in State Oriented Domains. The Sixteenth European Conference on Artificial Intelligence, Valencia, Spain (2004) 89-93
11. A.Agogino and K.Tumer. Multi Agent Reward Analysis for Learning in Noisy Domains. In Proceedings of the Fourth International Joint Conference on Autonomous Agents and Multi-Agent Systems, Utrecht, Netherlands (2005)
12. Sarvapali Dyanand Ramchurn. Multi-Agent Negotiation using Trust and Persuasion. PhD Thesis. Faculty of Engineering and Applied Science, School of Electronics and Computer Science, University of Southampton, Southampton, England (2005)

# A Frustum-Based Ocean Rendering Algorithm

Ho-Min Lee, Christian Anthony L. Go, and Won-Hyung Lee

Chung-Ang University,  
Department of Image Engineering,  
Graduate School of Advanced Imaging Science and Multimedia and Film,  
221 Hukseok-Dong, Dongjak-Gu, Seoul, Korea  
grancia@gmail.com, chipgo@gmail.com, whlee@cau.ac.kr

**Abstract.** Real-time water rendering has always posed a challenge to developers. Most algorithms concentrate on rendering small bodies of water such as pools and rivers. In this paper, we proposed a real-time rendering method for large water surfaces, such as oceans. This algorithm harnesses both the PC and GPU's processing power to deliver improved computing efficiency while, at the same time, realistically and efficiently simulating a large body of water. The frustum-based algorithm accomplishes this by representing a smooth water surface as a height value of the viewer, since surface size can be fluidly calculated given the camera frustum position. This algorithm has numerous potential applications in both the gaming and the movie industry. Experimental results show a marked improvement in computing power and increased realism in large surface areas.

## 1 Introduction

Together with the GPU's emergence in the last couple of years, real-time water rendering has likewise evolved. Early rendering techniques resulted in water surfaces that were not entirely realistic due to algorithm and processing power constraints. Today's GPU's however, with their increased computing power, afford developers added flexibility to render large bodies of water in real time.

In order to create realistic large water surfaces in real-time rendering, three components need to be addressed:

1. **Wave generation:** The height value of the water surface should be calculated in real-time, because waves cause the ocean to have a dynamic surface[1].
2. **Reflection and Refraction:** The overall look of water is based on reflected and refracted light. Both reflected and refracted light differ depending on the angle the surface viewed at.
3. **Water surface size:** A larger scale rendering must be done, taking into account height and point of view since the ocean represents a large body of water.

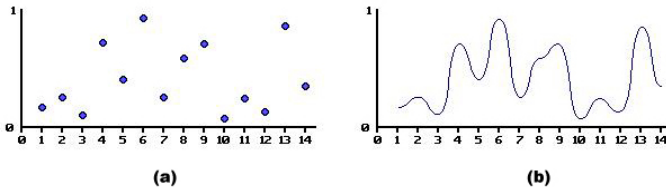
Among the three factors, for large-scale water rendering, we must primarily address surface size because the bigger the surface size, the more computing power needed to render it. Inefficient rendering will result in decreased performance and an overall decrease in speed when applied to games. This paper introduces an alternative, efficient approach for realistic large-scale water rendering.

This paper is organized as follows. Previous works of the real-time water rendering are described in Section 2. In Section 3, the proposed algorithm and the experimental results are shown. The conclusions of our research are stated in Section 4.

## 2 Water Surface Representation

### 2.1 Expressing Body of Water as Volume with Perlin Noise

The two most common algorithms employed for water surface generation are the Navier-Stokes and Perlin Noise Equations. While the Navier-Stokes Equation is the most commonly implemented of the two[7], it is, however, limited by its inability to efficiently render large bodies of water due to its high computational requirements[2]. Thus, to model large bodies of water, this paper utilizes the Perlin Noise Equation which creates a continuous noise. Two functions are necessary in order to create a Perlin Noise, a noise function, and an interpolation function[5]. A noise function is simply a seeded random number generator which takes an integer as a parameter, and returns random number based on that parameter. A continuous function can then be defined by interpolating the parameters. Fig. 1 graphically illustrates the processing[3]. Perlin noise can be



**Fig. 1.** (a) Demonstration of random numbers in coordinates (b) Demonstration of interpolated curve using random numbers

created using noise functions. When diverse noise functions are combined, a new noise pattern can be achieved.

### 2.2 Reflection and Refraction

When light strikes a water surface, it is both reflected from and refracted into the surface. To realistically replicate this, we employ Shader programming HLSL (GPU). Fig. 2 Shows reflected light and refracted light on a water surface.

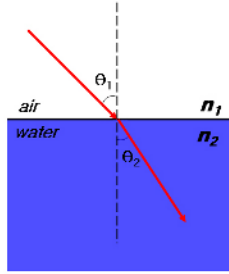


Fig. 2. Illustration of reflected and refracted light by Snell's law

Equation (1) demonstrates Snell's law which is used to calculate for reflection and refraction on water surface. We get  $\theta_1$  and  $\theta_2$  from the equation, and we can get realistic water surface which is applied reflected light and refracted light.

$$\frac{\sin\theta_1}{\sin\theta_2} = \frac{n_2}{n_1} \tag{1}$$

### 3 Frustum-Based Algorithm

#### 3.1 Water Surface Size

In real-time water rendering, a large water surface requires high processing power[8]. To reduce processing requirements and increase rendering efficiency for such large surfaces, we use shader programming and propose a new algorithm. This algorithm improves the projected grid algorithm by Claes Johanson[4]. The projected grid algorithm is sufficient when the water surface is viewed from a low viewpoint, however the water surface is not correctly displayed when the viewpoint (i.e. height value) of a viewer is increases. As a result, the water surface appears cut and looks smaller than it should. This is a serious problem when expressing a large water surface. Moreover, speed is an issue as it is not feasible to apply it to games which use about 30 fps.

#### 3.2 Render Water Surface

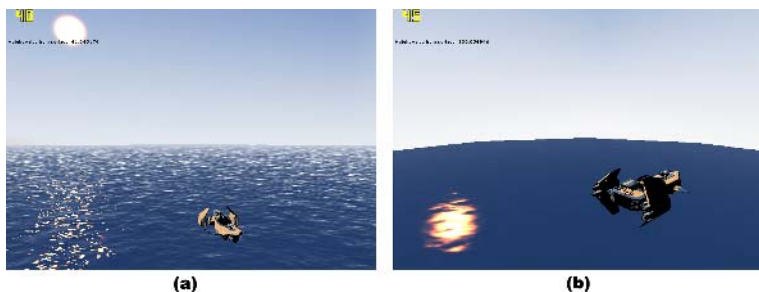
The water surface is rendered within the rendering frustum and surface size is defined by viewer's frustum which is a function of the rendering frustum. Equations (2) and (3) describe how the water surface is defined using viewer height value.

$$W_{range} = V_{HF} \cdot \sin\frac{3}{2}\theta \cdot \min(W_{range}), \quad (\theta < 60) \tag{2}$$

$$W_{range} = \max(W_{range}). \quad (\theta > 60) \tag{3}$$

Where  $V_{HF}$  is orthogonal height value from the water surface and  $\theta$  represents the angle between the surface and the user viewpoint. When viewing the

ocean from a beach, minimal water surface can be seen on the horizon. The  $\min(W_{range})$  value represents this water surface size. However, as we increase viewpoint height from a beach to that of an airplane, the ocean surface consequently increases on the horizon. After a certain vertical threshold, increases in height will no longer have any effect on surface area on the horizon.  $\max(W_{range})$  value denotes this limit on surface size. Fig 3. illustrates the water surface as height value, Fig. 3(a) is screenshot of the water surface with a height value of 60, while Fig. 3(b) is screenshot with a height value parameter of 800. Although,



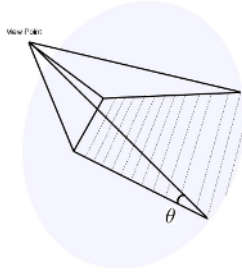
**Fig. 3.** Water surface rendering as height value

rendering size is substantially increased, our algorithm demonstrates efficient rendering with a refresh rate of approximate 40fps. This is true for both micro and macro views of the rendered ocean. An increased efficiency in grid calculation results in a significant improvement in rendering speed. The traditional projected grid algorithm creates a grid size of 128 by 256[4], conversely, our improved algorithm calculates grid size as a binding of neighboring grid coordinates, thus resulting in improved speed. Equation (4) shows how the grid coordinates bind with neighboring grid coordinates.

$$G_{bind} = \sum_{i=m} \sum_{j=n} \frac{\vec{c}_{i,j} + \vec{c}_{i+1,j+1}}{2} \quad (4)$$

### 3.3 Definition of Water Rendering Size

In our algorithm, we utilize two frustums for water rendering, one for the viewer and the other for the water body. Rendering of the body of water is done in detail within the frustum which is defined in post-perspective space. Areas outside of the camera frustum are not rendered. This is to minimize processing overhead. Fig. 4 shows how a surface can be presented in post-perspective. As described in Fig. 4, the water surface is rendered by calculating the transposition area of the camera frustum and surface.  $\theta$  which is described in Equation (2), (3) is the angle of the water surface and the camera viewpoint. And although the size of rendering surface is getting bigger as viewpoint and  $\theta$ , this algorithm shows efficient computing power because we utilize the LOD algorithm[9].

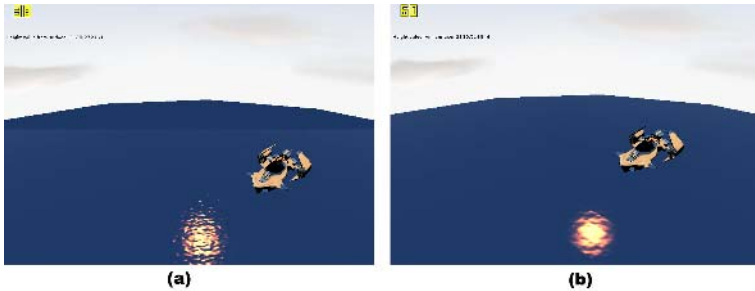


**Fig. 4.** Define a range for water rendering in post-perspective space

### 3.4 Experimental Results

We use common factors to achieve a realistic, believable water surface, these are reflection, refraction and waves. We, at the same time, minimize processing overhead to facilitate the rendering of a large mass of water. Fig 5. shows an ocean rendering using our algorithm. A 40% improvement in computing efficiency was achieved. Experiments were performed on a Geforce 6600 GPU with Pentium 4 processor running at 3.0 GHz. In this environment, we demonstrate enough efficiency to apply this method of water simulation to a game.

Comparing the two algorithms, Fig. 5(a) presents the projected grid algorithm and Fig. 5(b) is our algorithm. Both of these use the same height value of 2130, but the projected grid algorithm cuts the water surface and displays a part of earth surface, because it can no longer be rendered. We can clearly distinguish the improvement over the traditional method.



**Fig. 5.** (a) Rendering result by the projected grid algorithm (approximate height value = 2130) (b) Rendering result by our algorithm (approximate height value= 2130)

## 4 Conclusion

We have presented an efficient method for large water surface rendering using a frustum-based algorithm. Realistic water rendering has always posed a challenge



to developers. The tradeoff was always a compromise between computing power and size of the body of water. Realistic simulations are no longer the exception but have become the norm with the continuing advances in technology. Our algorithm addresses these demands by giving the developer the freedom to model a large body of water such as an ocean without having to worry about impossibly high computational demands.

## Acknowledgment

This research was supported by the Korea Culture and Contents Agency at the Culture Contents Technology Venture Center.

## References

1. Simon Premoze, Michael Ashikhmin.: Rendering Natural Waters. IEEE Computer Society. (2000) 23
2. Jim X. Chen , Niels da Vitoria Lobo.: Toward interactive-rate simulation of fluids with moving obstacles using Navier-Stokes equations. *Graphical Models and Image Processing*. **57** (1995) 107–116
3. David S. Ebert, F. Kenton Musgrave, Darwyn Peachey, Ken Perlin, Steven Worley.: *Texturing & Modeling - A Procedural Approach* Second Edition. AP Professional. (1998)
4. Claes Johanson.: Real-time water rendering. Lund University. (2004)
5. K. Perlin.: An image synthesizer. In B. A. Barsky (ed). *Computer Graphics (SIGGRAPH '85 Proceedings)*. **19** (1985) 287–296
6. John C. hart.: Perlin noise pixel shaders. *ACM Trans Graphics(SIGGRAPH Proc)* (2001) 87–94
7. James F. O'Brien, Jessica K, Hodgins.: Dynamic Simulation of Splashing Fluids. *Proceedings of computer Animation*. (1995) 198–205
8. Ping Wang, Daniel S. Katz, Yi Chao.: Optimization of a parallel ocean general circulation model. *Proceedings of ACM on Supercomputing*. (1997) 1–11
9. BLOW, J.: Terrain rendering at high levels of detail. *Game Developers Conference*. (2000)

# A Model of Video Coding Based on Multi-agent\*

Yang Tao<sup>1</sup>, Zhiming Liu<sup>2</sup>, and Yuxing Peng<sup>1</sup>

<sup>1</sup> School of Computer Science

National University of Defense Technology, Changsha, China

<sup>2</sup> Department of Computer, Nanhua University, Hengyang, China  
tao\_yang76\_8@hotmail.com

**Abstract.** In this paper we first propose a model of video coding based on multi-agent systems to improve the coding efficiency for H.264. We adopt the scheme of MAS in which the frame agent is designed to get information from the encoded frames regarding which reference macroblocks to select and to find the best motion vector by intercommunicating to each other and the whole coding process can be executed in a parallel way. Each frame agent can do himself coding work, and Motion Estimation can be used through the intercommunication between all the limited frame agents. In addition, a variety of Agents constructions and the ways to implementation are also discussed. The analysis study show that our design model is a valid and the proposed model presents us a novel video coding technology compared to other classical methods, which is a kind of technology fusion of signal processing and AI.

## 1 Introduction

The new JVT (or H.264, MPEG-4 AVC) [1] video-coding standard has gained more and more attention recently. The encoder complexity, however, is greatly increased, which makes it difficult for practical applications. Among these features, enhanced Inter and Intra prediction techniques are key factors to the success of H.264. Several attempts have been made to explore fast algorithms in intra-prediction mode decision and in inter-prediction for H.264 video coding [2-5], but there are still more efforts that we can do to achieve a better encoding efficiency. We can do some attempts to improve the coding efficiency for H.264 based on multi-agent technology.

Designing a practical multi-agent system should consider the fundamental characteristics of agent technology itself and video coding scheme. Intelligent agents have informational as well as motivational attitudes, such as observing, communicating, revising beliefs, planning, and committing [6-8]. Intelligent agents mainly have many aspects as stated in [6]. In artificial intelligence research, agent-based systems technology has been hailed as a new paradigm for conceptualizing, designing, and implementing software systems. Agents are sophisticated computer programs that act autonomously on behalf of their users, across open and distributed environments, to solve a growing number of complex problems. Increasingly, however, applications require multiple agents that can work together [7-8]. A multi-agent system (MAS) is a loosely coupled network of software agents that interact to solve problems that are

---

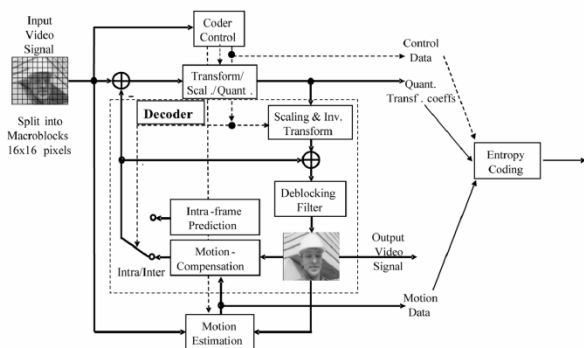
\* The research work was supported by NSF project no. 60433040.

beyond the individual capacities or knowledge of each problem solver. Hence requires us to do our endeavor to make use of the MAS to video coding system to improve coding efficiency as much as possible.

The rest of the paper is organized as follows: in the Section 2, we state the frame agent and its related definitions. Section 3 provides detail of the model of video coding based on the MAS. Section 4 presents the analysis study and discussion. Section 5 discusses further related work. Finally, we present our conclusions in Section 6.

## 2 Frame Agent

In H.264, the VCL design follows the so-called block-based hybrid video coding approach (as depicted in Fig.1), in which each coded picture is represented in block-shaped units of associated luma and chroma samples called macroblocks. The basic source-coding algorithm is a hybrid of inter-picture prediction to exploit temporal statistical dependencies and transform coding of the prediction residual to exploit spatial statistical dependencies. According to the JVT reference software JM10.1 [9], the encoder of H.264 processes frames one by one. So a frame can be regarded as a single entity. Then the frame agent can be described.



**Fig. 1.** Basic coding structure for H.264/AVC for a macroblock

**Definition 1.** *Frame Agent is a kind of abstract autonomous agent, which can do the encoding job for a frame, especially do the intra-prediction job of the encoder alone.*

Given one frame, a frame agent splits the frame into macroblocks; the agent encodes macroblocks one by one. A frame agent always has tasks including spatial prediction for intra-coding, block transforms, entropy coding and so on. Sometimes a frame agent has to interact with other agents providing necessary information to implement the motion estimation and motion compensation technology.

**Definition 2.** *I-Frame Agent is a frame agent, which performs a given task to encode an I-frame or I-slice using information gleaned from its environment to act in a suitable manner so as to complete the task successfully.*

Given an I-frame or I-slice, an I-frame agent can complete its task alone without any other peer agent's help. In addition, an I-frame agent can provide some information to other agents, such as P-frame agents and B-frame agents.

**Definition 3.** *P-Frame Agent is a frame agent, which performs a given task to encode a P-frame or P-slice using information gleaned from its environment to act in a suitable manner so as to complete the task successfully.*

Given a P-frame or P-slice, a P-frame agent can complete its task using information gleaned from its environment including other I-frame agents of P-frame agents. At the same time, the P-frame also provides the information of motion vector and corresponding residual block to other agents, such as P-frame agents and B-frame agents.

**Definition 4.** *B-Frame Agent is a frame agent, which performs a given task to encode a B-frame or B-slice using information gleaned from its environment to act in a suitable manner so as to complete the task successfully.*

Given a B-frame or B-slice, a B-frame agent can complete its task using information gleaned from its environment including other I-frame agents of P-frame agents. In addition, it should have the ability to justify which information is the exact it need. It is more intelligent than I-frame agents or P-frame agents.

### 3 The Proposed Video Coding Model

#### 3.1 Input Agent

**Definition 5.** *Input Agent is an intelligent agent, which performs a given task to prepare for the environment for other agents using information gleaned from system users to act in a suitable manner so as to complete the task successfully.*

Given a video sequence and configure files, an input agent can get all the frames data from the input video, and all the parameters for the encoder environment from configure files and distribute them to corresponding frame agents in order.

#### 3.2 Output Agent

**Definition 6.** *Output Agent is an intelligent agent, which will finish the whole coding process for H.264 and output the compressed data using information gleaned from other agents to act in a suitable manner so as to complete the task successfully.*

Before frame agents leave out of the MAS, they will send the compressed video data to the output agent. The output agent does the last process of the encoder work:

- Get information from the input agent to build the output environment.
- Glean all coded data information from frame agents and output the data.
- Inform everyone that the whole process is finished and clean the environment.

### 3.3 MAS and the Proposed Model of Video Coding

The study of MultiAgent Systems (MAS) focuses on systems in which many intelligent agents interact with each other. The agents are considered to be autonomous entities, such as frame agents. Their interactions can be either cooperative or selfish.

MAS researchers draw on ideas from many disciplines outside of AI. MAS are inspired by models from biology (ecosystems) and economics (markets). Many researchers point the most important theoretical concepts and practical issues associated with the design of intelligent agents and Multi-agent systems [10]. We present the process of the model as follows:

- The User prepares the whole coding task and then submits them to the input agent to build the encoding environment.
- The input agent confirms each frame's type and then distributes it to a corresponding frame agent.
- Each frame agent gets frame data encoded after its task finishes.
- At the end, the output agent gleans all coded data information and output the data in a regular format.

Suppose the total sum of frames  $N=V \cdot G$ , where  $V$ s and  $G$  represent the number of GOP and the number of pictures in a GOP respectively. So for each GOP, the input agent distributes pictures to frame agents correspondingly in the MAS. When one GOP is done, the input agent goes on processing another one until all GOPs finish.

A picture is partitioned into fixed-size macroblocks. Macroblocks are the basic building blocks of the standard for which the decoding process is specified. The basic coding algorithm for a macroblock is still adopted in our proposed model. The work is done in frame agents, including exploiting the spatial correlation between the adjacent macroblocks and the temporal correlation between frames as illustrated in Fig.2.

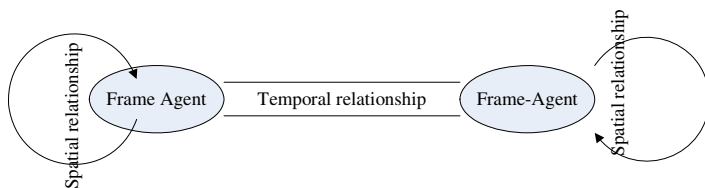


Fig. 2. The relationship between Frame Agents

## 4 Analysis Study

### 4.1 The Implement for Intra-coding Technology in Our Model

Intra prediction in H.264 exploits the spatial correlation between the adjacent macroblocks. In the proposed model, frame agents do all the task of intra-coding. So it is still the key problem to explore fast algorithms in intra-prediction mode decision for H.264 video coding. The algorithms appear as software components in the frame agents system.

## 4.2 The Implement for Motion Estimation Technology in Our Model

Motion Estimation has been a hot issue since JVT meeting started. Several valuable Fast Motion Estimation algorithms [5] were proposed and a great portion of ME time reduction (up to more than 90%) and total JM encoding time was reduced by 60% on average. To qualify the speed accelerated by Fast Motion Estimation, frame agents should bare in mind that how much PSNR loss or bit rate increasing fast algorithms cost. Thus, frame agents can get the best motion vector and better coding efficiency with fast motion estimation algorithms in the MAS.

## 4.3 The Performance of Proposed Model

In MAS research there are investigations of many properties and autonomous behaviors of agents, but system-level interoperability and autonomy are the behaviors of direct relevance to the system.

In the proposed model, the input agent has the ability to integrate multiple frame agents to provide the “runtime integration”. All the frame agents are autonomous and intelligent, and the abilities to collaboratively perform a task (e.g. encode a video sequence) based on the exchange of meaningful information.

Another reason for using MAS technology in the video coding is that multi-agent systems presume a common abstract architecture of functional services that can be implemented in heterogeneous ways. This facilitates the integration of a myriad of disparate software systems and components. So intelligent frame agents can easily find the best motion vector by intercommunicating to each other and the whole coding process can be executed in a parallel way in the proposed model.

So the proposed model is valid and efficient.

## 5 Further Related Researches

Now we present some important work as follows:

- **Autonomy:** as mentioned above, only the input agent needs some initialization by humans or other systems and frame agents are autonomous to do their tasks. So in order to finish their tasks efficiently, a better design for agents will be considered in the further work in the model of video coding.
- **Intelligence:** as our model is different to the classical video coding systems. So much more intra-coding and motion estimation algorithms need to be improved or proposed.
- **Protocols:** if we have a better protocol for frame agents to intercommunicate to each other, the whole process will go on smoothly and the model will get a better coding efficiency. In the further work we have to do much more with the interaction protocols between agents in the MAS.

## 6 Conclusion

In this paper, we have taken multi-agent technologies for designing a video coding model for H.264. We first present the frame agent, which is an efficient agent

working with the encoder of H.264. Then propose the video-coding model with many frame agents and other agents. Through analysis, we evaluate the performance of the frame agent and the novel video coding model. As part of future work, we would apply our research to design and implement a distributed protocol for frame agents to intercommunicate, so as to make it more efficient in video coding for H.264.

## References

1. H.264, Draft ITU-T Recommendation and Final Draft International Standard, Pattaya, Thailand, 2003
2. Pan, F.; Rahardja, L.S.; Lim, K.P.; Wu, L.D.; Wu, W.S.; Zhu, C.; Ye, W.; Liang, Z.; Fast intra mode decision algorithm for H.264-AVC video coding; Image Processing, 2004. ICIP '04. 2004 International Conference on Volume 2, 24-27 Oct. 2004 Page(s): 781 - 784 Vol.2
3. Chen Chen; Ping-Hao Wu; Homer Chen; Transform-Domain Intra Prediction for H.264; Circuits and Systems, 2005. ISCAS 2005. IEEE International Symposium on 23-26 May 2005 Page(s): 1497 – 1500
4. Efficient block-size selection algorithm for inter-frame coding in H.264/MPEG-4 AVC, Yu, A.C. Acoustics, Speech, and Signal Processing, 2004. Proceedings. (ICASSP '04). IEEE International Conference on, Vol.3, Iss. , 17-21 May 2004 Pages: iii- 169-72 vol.3
5. Xiaoquan Yi, Jun Zhang, Nam Ling, and Weijia Shang, "Improved and simplified fast motion estimation for JM", JVT-P021, July, 2005
6. Bradshaw, J.M. An Introduction to Software Agents. In: Software Agents, J.M. Bradshaw (Ed.), Menlo Park, Calif., AAAI Press, 1997, pages 3-46
7. D. Kinny, .The AGENTIS agent interaction model in Intelligent Agents V.Proc. Fifth Int. Workshop on Agent Theories, Architectures, and Languages (ATAL-98), Lecture Notes in Artificial Intelligence, J. P. M;ßuller, M. P. Singh, and A. S. Rao (Eds.), Springer-Verlag: Heidelberg, 1999.
8. Bema Koes M,Nourbakhsh I,Sycara K. Communication efficiency in multi-agent systems[C].In :Robotics and Automation,Proceedings ICRA'04.2004 IEEE International Conference on,2004
9. Joint Video Team (JVT), reference software JM10.1, <http://iphome.hhi.de/~suehring/tml/download/jm10.1.zip>
10. Dependable Agent Systems. IEEE Intelligent Systems Special Issue, Volume 19, Number 5; September/October 2004.

# PDC-Agent Enabled Autonomic Computing: A Theory of Autonomous Service Composition

Bei-shui Liao<sup>1</sup>, Li Jin<sup>1</sup>, and Ji Gao<sup>2</sup>

<sup>1</sup> Research Center of Language and Cognition, Zhejiang University,  
310028 Hangzhou, China

baiseliao@zju.edu.cn, jinli\_pearl@zju.edu.cn

<sup>2</sup> Institute of Artificial Intelligence, Zhejiang University,

310027 Hangzhou, China

gaoji@mail.hz.zj.cn

**Abstract.** In order to release the management burden during the process of dynamic sharing and integration of heterogeneous web applications, this paper proposes a framework of PDC-agent enabled autonomic computing (PEAU) system, in which PDC-agent is a novel agent that integrates policy-based management theory and contracts-based cooperation mechanism into traditional BDI agent model. The P, D, and C are the abbreviation of policies, desires and contracts respectively. The PEAU system is designed to realize the properties of autonomic computing, including self-configuration, self-optimization, self-healing, and self-protection, etc. This paper focuses on the autonomous VO (virtual organization) formation and service composition.

## 1 Introduction

Currently, it has been a research hotspot in the areas of distributed systems to enable the process of sharing and integration of heterogeneous web applications to realize self-management (autonomic computing [1]), with the guidance of dynamic management requirements. In recent three years, a lot of emerging methods and technologies have been proposed to realize the characteristics of autonomic computing. Among them, agent-based methods [2,3] and policy-based ones [4,5] are dominant. The former adopts intelligent agents as autonomic elements. Since agents are characteristic of reactivity, pro-activity, autonomy and sociality, they are able to autonomously manage local resources and take part in cooperation with other agents. On the other hand, by separating management logic from application logic of a system, policy-based methods can dynamically regulate the behaviors of system components without changing code, i.e., policies provide an efficient approach to reflect dynamic management requirements. However, these two kinds of methods have their disadvantages. First, traditional agent technology [6] is lack of mechanism to reflect high-level management requirements that are possessed by policy-based methods. Second, although current policy-based methods [7] are able to represent high-level business requirements and dynamically manage the behaviors of underlying components, the “event-condition-action” policy representation languages and management patterns result in the fact that policy-based management systems can only perform basic



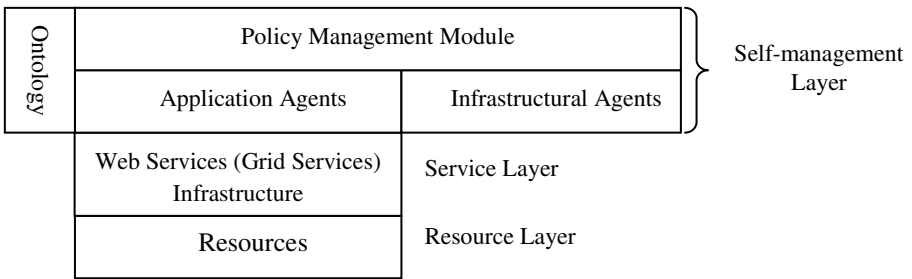
actions according to the trigger and constraint of a policy, without the capability of analysis, reasoning and decision-making. This is just the main functionality of an agent.

So, in our previous work [8,9], we have presented a novel agent that integrates policy-based management theory and contracts-based cooperation mechanism into traditional BDI agent model, called PDC-agent. The P, D, and C are the abbreviation of policies, desires and contracts respectively. On the basis of PDC-agent, this paper proposed a framework of **PDC-agent Enabled Autonomic computing (PEAU)** system, which is designed to realize the properties of autonomic computing, including self-configuration, self-optimization, self-healing, and self-protection, etc. This paper focuses on the autonomous VO (virtual organization) formation and service composition.

The structure of this paper is organized as follows. In section 2, a general framework of the PEAU system and its working principle are proposed. Then, in section 3, we introduce the theory of PDC-agents enabled VO formation and service composition. Finally, in section 4, conclusions are drawn.

## 2 The Architecture of PEAU System and Its Working Principle

The architecture of PEAU system is composed of three layers, including resource layer, service layer and self-management layer, as shown in Fig.1.



**Fig. 1.** The architecture of PEAU system

The resource layer contains various resources, including computation, storage, software applications and database, etc. At the service layer, web service (grid service) infrastructure provides standard interfaces and protocols, which hide the heterogeneity of the underlying components. The self-management layer is the core of PEAU system. In this layer, various agents guided by policies constitute federated organizations, called agent federations (AFs). AFs can be nested according to specific application requirements. An agent federation is composed of a federation management agent (FMA), several local application agents, several external role enacting agents (REAs), and several local infrastructural agents such as trading agents (TrA), monitoring agents (MntA), etc. FMA is in charge of the formation of the logical structure of a VO, the recruitment of member agents, and the organization of system cooperation. During the process of VO formation, FMA (when necessary) may invoke trading agents (TrA) to look for REAs via mediate agents (MedA), negotiate with

them, and sign contracts if agreements are reached. The external REAs are those service providers who can provide required services. When REAs are prepared, FMA firstly composes sub-services into more coarsely granular services, and then schedules the REAs to perform the assigned sub-services respectively. Meanwhile, when sub-services are under execution, MntAs are invoked to monitor the states of service interaction. If some exceptions arise, the FMA initiates exception treating mechanism to treat with the exceptions. Specifically, if the exceptions result from the violation of contracts, the measurement of the punishment will be taken under the surveillance of authority agent (AthA).

The various kinds of agents mentioned above are directed by policies that are specified by human administrators. Policies can be categorized as high-level policies, concrete policies, and enforceable policies [10]. The specification of these kinds of policies and the refinement mechanism that automatically refines high-level policies to concrete/enforceable policies are not presented in this paper. Readers are referred to [10,11] for details.

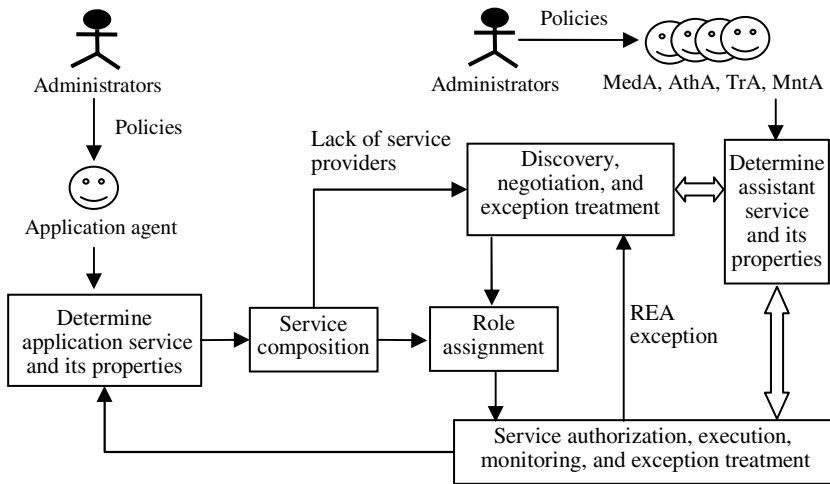


Fig. 2. The working principle of PEAU system

The working principle of PEAU system can be illustrated as Fig.2. An application agent initiates a process of VO formation and service composition according to the specific business objectives. On the one hand, for the intention of providing internal services to potential consumers, the application agent may invoke trading agent (TrA) to advertise her service-providing abilities to a certain mediate agent (MedA). On the other hand, on the basis of the status of internal resources and real-time business requirements, she makes decisions about whether she should outsource some services from external service providers. If so, she invokes a trading agent (TrA). The latter looks for potential service providers via a MedA. After the MedA finds some qualified service providers, she sends the related information to the TrA. Then, TrA negotiates with the qualified service providers in sequence. If an agreement is reached, she will sign a contract with her counterpart. When all service providers are prepared, the

application agent (acting as FMA) initiates the process of service execution. If there are some exceptions during the VO formation and service execution, the application agent will cope with the exceptions with the support of infrastructural agents.

The above VO formation, service composition, service exception and exception treatment form a control loop. In this loop, each participating agent first chooses the preferred service as her next goal. Then, according to this goal, she produces an intention to perform the service. During these processes, agent circularly determines *what she will do next* and *how she does it*, at each time point. In Fig.2, *determining next executing service and its properties* resolves the problem of *what she will do next*; while service composition and execution deal with *how she does it*.

Within PEAU system, agents act as autonomic elements. Behaviors of these agents not only depend on the variation of environment, but also on the requirements of their users and other agents respectively. In other words, in PEAU system, agent's goals are determined by three factors, including agent's internal desires, and obligations that are brought about by policies and contracts respectively. Therefore, the traditional BDI agent is not feasible in that its decision-making is mainly based on its own desires. We extend the traditional BDI agent model with obligations arisen from policies (PObligations for short) and obligations from contracts (CObligations for short), called BGI<sub>PDC</sub> logic. BGI<sub>PDC</sub> logic and PDC-agent model are formulated in [9,12].

### 3 PDC-Agents Enabled VO Formation and Service Composition

As shown in fig.2, according to the policies specified by its administrator, PDC-agent makes decisions at each time point about the service to be executed next. Initially, PDC-agent's state is *waiting*. When new motivations (PObligations, Desires, and/or CObligations) arise, PDC-agent treats with the possible motivation conflicts and determines the next executing service and its properties (goal generation). Then, she checks the type of the service. If the service is a complex service, compose it and get a composition scheme. And then, according to the complex service (or simple service) and its properties, she checks whether the capabilities of the existing service providers are enough to fulfill the complex service with specific properties. If the existing providers can't meet the requirements, the PDC-agent invokes its local trading agents (TrAs), according to the service acquiring policies and service trading policies, to look for required service providers through discovery and negotiation, and sign contracts with them if the negotiation is successful. If the process of service discovery and negotiation fails, she treats with the organization exceptions. Finally, when service providers are prepared, she schedules each sub-service to be executed according to the resource usage policies (and the service composition scheme).

The above PDC-agent who manages the process of VO formation and service composition is called federation management agent (FMA), as mention in section 2. Meanwhile, during the execution process of application service, the FMA invokes the trading agent to monitor the process of service execution. If exceptions arise, an exception treatment program will be invoked.

The mechanisms of service discovery, negotiation, and monitoring are not presented in this paper. Readers are referred to [13,14] for details.

In PEAU system, service composition is realized by means of a service composition plan. The service composition plan decomposes a complex service into a set of atomic services (web services or grid services) and/or a set of finer agent services. The decomposition is nested, until all sub-services can be performed by invoking domain services or agent services. According to different conditions, a complex service can be decomposed into different sets of sub-services. In this paper, it is called a reduction scheme (RS), which is defined in the form of EBNF as follows.

$$RS ::= \{ \text{'(}' \leftarrow \text{'(}' (\mathbf{SubService} \mid \text{'Sequence('} \mid \text{'Concurrence('} \{ \text{'('} \leftarrow \mathbf{SubService} \mathbf{Condition} \text{'')'')'')'')'')' \}$$

Based on the reduction schemes, the service composition plan is designed according to domain-specific requirements. The detailed definition of PDC-agent' plan is referred to [8] and not presented in this paper.

## 4 Conclusions

This paper proposed a model of PDC-Agent Enabled Autonomic Computing (PEAU) system, based on which a theory of autonomous VO formation and service composition was presented. The PEAU system is founded on the service-oriented architecture and its supporting technologies, including web service and grid service infrastructure. So, in this model, resources are abstracted as services, which are the management objects of PDC-agents. As autonomic elements, PDC-agents take part in cooperation by the guidance of high-level business strategies (represented as policies) to realize service discovery, negotiation, composition, process monitoring, and exception treating, etc. Cooperation relationships are established and operationalized by means of contracts, which make the cooperation among autonomic elements more stable and more reliable.

Characteristics of PEAU system and the mechanism of PDC-agent enabled VO formation and service composition mainly include the following three aspects. First, the process of VO formation and service composition is autonomously managed by PDC-agents; second, human administrators are able to dynamically guide the behaviors of PDC-agents in terms of policies, without modifying the codes of underlying components; third, service composition is based on composition scheme, which are based on the composition plans and the real-time information. The composition scheme is produced at run time, so it is adaptable to the dynamic VO environments.

## Acknowledgements

We gratefully acknowledge the support of the National Grand Fundamental Research 973 Program of China under Grant No.2003CB317000, and the 985 Project of Zhejiang University, China.

## References

1. Jeff O. Kephart and David M. Chess. The Vision of Autonomic Computing[J]. *IEEE Computer*, 2003:36(1): 41~50.
2. Dario Bonino, Alessio Bosca, et al. An Agent Based Autonomic Semantic Platform [A]. In: *Proceedings of the International Conference on Autonomic Computing*, 2004, 189~196.
3. Huaglory Tianfield. Multi-Agent Autonomic Architecture and Its Application in E-Medicine [A]. In: *Proceedings of IAT'03, Halifax, Canada*, 2003, 601~604.
4. Sandeep Uttamchandani, et al. Eos: An Approach of Using Behavior Implications for Policy-Based Self-Mamanement[A], LNCS2867, 2003,16~27.
5. Murthy Devarakonda, et al. Policy-Based Autonomic Storage Allocation [A], LNCS2867, 2003,143~154.
6. M. Wooldridge, N.R.Jennings: *Agent Theories, Architectures, and Languages: A Survey* [A].In: *Proceedings of ECAI, Amsterdam, The Netherlands*, 1994. 1~32.
7. G. Tonti, J.M. Bradshaw, et al.Semantic Web Languages for Policy Representation and Reasoning: A Comparison of KAoS, Rei, and Ponder. In: *Proceedings of ISWC2003*, 2003.
8. Bei-shui Liao. *Service-Oriented Autonomic Computing Based on PDC-Agent* [D]. Hangzhou: Zhejiang University, 2006.3 (in Chinese).
9. Bei-shui Liao, Ji gao.A Model of Multi-agent System Based on Policies and Contracts [A].In: *Proceedings of CEEMAS'05* [C], LNAI3690, 2005: 62-71.
10. Bei-shui Liao, Ji gao, et al. *Ontology-Based Conceptual Modeling of Policy-Driven Control Framework: Oriented to Multi-Agent System for Web Services Management* [A]. In: *Proceedings of AWCC2004*[C], LNCS, 2004: 346-356.
11. Bei-shui Liao, Ji gao.An Automatic Policy Refinement Mechanism for Policy-driven Grid Service Systems[A]. In: *Proceedings of GCC2005*[C], LNCS3795, 2005: 166-171.
12. Bei-shui Liao, Hua-xin Huang, Li, Jin, Ji Gao. An Extended BDI Agent with Policies and Contracts. In: *Proceedings of PRIMA2006*.
13. Zhou Bin. *The Systematism of Assistant Service for Agents* (pp.9-24) [D].Hangzhou: Zhejiang University, 2004 (in Chinese).
14. Ji Gao, Cheng-xiang Yuan, Jing Wang. SASA5: A Method System for Supporting Agent Social Activities. *Chinese Journal of Computers*, 2005, 28(5): 1-11.

# QoS Based Routing in Wireless Sensor Network with Particle Swarm Optimization

Xi-huang Zhang and Wen-bo Xu

School of Information Engineering, Southern Yangtze University  
214122 Wuxi, Jiangsu, P.R. China  
zxhsytu@sytu.edu.cn, xwb@sytu.edu.cn

**Abstract.** There are many quality of service (QoS) challenges faced during wireless sensor networks (WSN) application. Many QoS metrics, not only data packet delay and bandwidth efficiency, but power consumption should be considered in network design. QoS metric levels are greatly affected by network routing. Since the routing solution space grows exponentially with the size of the network, it is necessary to research efficient combinatorial optimization algorithms for routing. After studying intelligent particle swarm optimization (PSO) algorithm, a new routing algorithms based on PSO is described, which has the potential to address many QoS metrics together and has an outstanding searching ability. The approach is well founded theoretically as well as detailed algorithmically. Simulations and comparisons to some typical QoS routine methods show that particle swarm optimization based routing algorithm is effective.

## 1 Introduction

The rapid speed of wireless sensor networks (WSN) innovation has resulted in many new application fields. If we implement some works with WSN, the quality of service (QoS) should reach an acceptable and reasonable level. To improve QoS become a major problem in WSN design. Different network routing will cause the different QoS level. There are many approaches to maintain QoS[1,2]. Also a better routing algorithm should have these attractive features: autonomy, robustness and fault-tolerance. It is a NP problem to search a path that meets the need of QoS in WSN. It is true that the number of routing path is countless. When the number of nodes in wireless sensor networks rises, routing path solution space grows exponentially. It is necessary to use efficient optimization algorithms.

Recently the particle swarm optimization (PSO) becomes the one of the evolutionary computation techniques. The PSO was used to search optimal or near-optimal solutions in large search spaces [3]. The technique involves simulating social behavior among individuals (particles) “flying” through a multidimensional search space, each particle representing a single intersection of all search dimensions. The particles evaluate their positions relative to a goal (fitness) at every iteration, and particles in a local neighborhood share memories of their “best” positions. Then those memories are used to adjust their own velocities, and thus subsequent positions. In this paper, every routing path is expressed as individuals (particles). The best position of

particle is the goal of the routing path. The evaluation of the particle is the value of target function for QoS level.

The rest paper is organized as follows. In Section 2, we will introduce the basic concept of PSO and its usage. The wireless data networks routing is detailed in Section 3. In section 4, QoS metrics are discussed and at last, experiments and conclusions are given.

## 2 Basic Concept of PSO

In  $D$ -dimensional space, particle  $i$  is represented as  $x_i=(x_{i1},x_{i2},\dots,x_{iD})$ . Each particle also maintains a memory of its previous best position,  $Pbest_i=(p_{i1},p_{i2},\dots,p_{iD})$ . Particle  $i$  moves at a velocity along each dimension, represented as  $V_i=(v_{i1},v_{i2},\dots,v_{iD})$ . The  $Pbest_i$  of the particle  $i$  and the best fitness in the group neighborhood, designated  $Gbest$ , are combined to adjust the velocity along each dimension. Velocity is then used to compute a new position for the particle  $i$ . The portion of the adjustment to the velocity influenced by the individual's previous best position ( $Pbest_i$ ) is considered the cognition component, and the portion influenced by the best in the neighborhood  $Gbest$  is the social component. It is obviously that the new position of particle  $i$  is computed iteratively.

With the addition of the inertia factor  $\omega$ , PSO iteration formulae are:

$$v_i = \omega v_i + \eta_1 \text{rand}() (Pbest_i - x_i) + \eta_2 \text{rand}() (Gbest - x_i) \quad (1)$$

$$x_i = x_i + v_i \quad (2)$$

Where  $v_i$  is the velocity of particle  $i$ .  $x_i$  is current position of particle  $i$ .  $Pbest_i$  is the best position of particle  $i$  and  $Gbest$  is the best position of the group.

Using the above equation (1), a certain velocity that gradually gets closer to  $Pbest_i$  and  $Gbest$  can be calculated.

PSO utilizes several searching particles. These searching particles gradually get close to the global optimal point using its  $Pbest_i$  and  $Gbest$ . Initial positions of  $Pbest_i$  and  $Gbest$  are different. However, using the different direction of  $Pbest_i$  and  $Gbest$ , all particles gradually get closer to the global optimum.

## 3 Routing in Wireless Sensor Networks

Routing explores all paths available between source node and destination node. The basic principle of PSO based routing algorithms is to use stochastic exploration for new path discovery. This stochastic property is achieved by using routing tables that assign QoS metrics to next-hops. Special agent follows a next-hop based on these evaluations of synthetic QoS metrics. Regular data packets always follow the next-hop with the highest evaluations of synthetic QoS metrics. In this paper, the evaluation of synthetic QoS metrics is same as target function of the routing. There is a simple relationship between them.

### 3.1 Routing Table

A sample routing table is given in Table 1, where the first column corresponds to destination nodes and the first row corresponds to neighbors of the sources node A. There are three elements in a routing table unit. One is the synthetic evaluation QoS metric from source node A to its neighbor nodes such as 0.4 for A to B. The next element is the synthetic QoS metrics of the path from source node A to the destination node X through B. The third element is the path trip-time of the path. To maintain a path, enough energy reserve on the path to destination node is needed. Here the path trip-time is used to express the energy reserve. Table 1 shows the source node A has three neighbors B, C and D. Data packets are send from source node A to destination nodes X, Y and Z respectively.

**Table 1.** Routing table in node A

Source node A	Neighbor node B			Neighbor node C			Neighbor node D		
Destination node X	0.4	2.5	240	0.5	2.9	450	0.3	1.8	780
Destination node Y	0.6	3.6	460	0.4	5.1	750	0.6	2.6	680
Destination node Z	0.4	2.8	910	0.7	6.4	304	0.2	5.2	420

How to update the routing tables is important. To get the information of packet transmission, tow agent is designed. One is forward agent and the other is backward agent. Routing tables are updated as the forward agent sends a packet from source node to destination node. Once it reaches its destination, each forward agent tell the traveling time information and other QoS metric parameter to the backward agent, which updates the routing tables as it traces the path of the forward agent in reverse. It is sure that routing tables are updated only the path from source to destination is established. In PSO iteration, every particle (path) moves with the modification of the routing table.

### 3.2 Routing Path Initialization

To find an optimal path from source node to destination node, initial paths are needed. Two agents are employed to search initial paths, one is forward agent and the other is backward agent. The forward agent moves forward to find and determine the next node of the path, create and maintain the routing table. At the same time, forward agent collects many information of the node in the path. Backward agent is created by destination node and moves back from destination node to source node after forward agent reach the destination node.

Every forward agent includes four fields, source address, destination address, serial number and hop counter, with an additional table saving information of all nodes in the path. If a node want to communicate with another node, it will create a new forward agent with address itself and broadcast to every neighbor node. When a node receives a forward agent, it should check the destination node address and every address of the node in the path. If the destination node address is same as the address the node self, the forward agent reach the destination node. The search process is terminated. If the address of a node in path is same as the address the node self, there is a loop in the path. The forward agent is canceled. It should be noticed that forward agent is broadcasted, so there will be many different route paths. These paths are the initial particle swarm.



A backward agent is created and initialed by the destination node when a forward agent reaches. Backward agent has the same structure as forward agent. It is sent back to source node alone with the path of the forward agent. After these backward agents reach to the source node, the initial routing table is completely formed. Routing table will be midfield with PSO iteration.

## 4 Routing Path Optimization by PSO

It is obviously that the process to find the best path from source node to destination node in WSN is the process to search an optimal target in all solution space, which is similar to the process of optimal solution searching by the cooperation of all particle swarm.

Paths (particle swarm) are obtained by forward agents and backward agents or by iteration of PSO algorithm and are expressed as  $\{x_0, x_1, x_2, \dots, x_i, \dots, x_n\}$ . Every path  $x_i$  is a potential best routing path. Note that the best  $x_i$  as  $pbest_i$  when  $x_i$  modified. The best path in all paths is expressed as  $gbest$ . Assume that  $x_i$  with  $k$  hops consists of a set of nodes  $\{n_0, n_1, n_2, n_3, \dots, n_s, \dots, n_k\}$ . The evaluation of  $x_i$  is calculated by the target function that could be used to estimate the evaluation of  $x_i$ , the level of QoS. To search a next place of the path (particle)  $x_i$  is to adjust the former path  $x_i$ . The path is modified by the  $v_i$  according to the formula (2). In fact  $v_i$  can not be expressed analytically in routing. So how to obtain the next  $x_i$  is the key of PSO algorithm in path researching.

According to the meaning of formula (1) and (2), the next path is getting closer to the  $pbest_i$ . Assume that  $pbest_i$  with  $K$  hops consists of  $\{np_0, np_1, np_2, \dots, np_s, \dots, np_K\}$ . Some of nodes in  $x_i$  are same as in  $pbest_i$ . It is reasonable when a node  $n_s$  in path  $x_i$  replaced by a different node  $np_s$  selected in  $pbest_i$ , path  $x_i$  is closer to  $pbest_i$ . The same method is also used for third part of formula (1) for  $gbest$ .

When  $n_s$  is replaced by  $np_s$ ,  $np_s$  should broadcast the forward agent. If  $n_{s-x}$  and  $n_{s+y}$  ( $x, y \geq 1$ ) receive the forward agent, the broken path is mended and the new path  $x_i$  is recreated. Otherwise we should select another node in  $pbest_i$  instead  $np_s$ , may be  $np_{s+1}$ .

The proposed routing algorithm using PSO is expressed as follows:

Step1. Source node creates and initializes a forward agent. Broadcast the forward agent and then ready receive a backward agent. A path called  $x_i$  is got when a backward agent is received. (Processes of forward agent and backward agent are omitted here)

Step2. While iteration number < maximum iteration number

Step3. {For every path  $x_i$  Do

Step4. { Calculate the target function (QoS metrics) for path  $x_i$ .

Step4. Update  $pbest_i$  when the evaluation of new  $x_i$  is better than  $pbest_i$ .

Update  $gbest$  if the evaluation of new  $x_i$  is better than  $gbest$ .

Step5. When the target is satisfied, go to step 9

Step6.  $x_i$  is iterated by above method}

Step7. New searching point (the better path) is got.

Step8. Iteration number increase 1 }

Step9. Check and determine the final path.

WSN is expressed as weighted directed graph  $G(V, E)$ ,  $V$  is a set of nodes in WSN. If there are  $m$  nodes in network,  $V = \{n_1, n_2, n_3, \dots, n_m\}$ . If node  $i$  and node  $j$  are neighbor

nodes, note  $l(i,j)$ . A path between node  $a$  and node  $b$  is  $P(a,b)$ .  $E$  is a set of paths.  $E = P(a,b) | \{a \in V, b \in V \text{ there is a path between } a \text{ and } b\}$ . The QoS metrics for  $l(i,j)$  are time delay  $D(i,j)$ , energy reserve  $E(i,j)$  and synthetic QoS metric. Synthetic QoS metric are such as Signal Noise Ratio (SNR) and Bandwidth Efficiency Ratio (BWER), expressed as  $S_{SNR}(i,j), S_{BWER}(i,j)$ . When  $P(a,b)$  is formed, QoS metrics for  $P(a,b)$  are expressed in (3).

$$\begin{aligned}
 D_{p(a,b)} &= \sum_{l(i,j) \in P(a,b)} D(i,j) \rightarrow \min < \text{deadline} \\
 E_{p(a,b)} &= \min_{l(i,j) \in P(a,b)} E(i,j) \rightarrow \max \\
 S_{p(a,b)} &= \sum_{l(i,j) \in P(a,b)} S_{SNR}(i,j) + S_{BWER}(i,j) + \dots \rightarrow \max > \text{system demand}
 \end{aligned}
 \tag{3}$$

[3,4,5,6,7] introduced how to calculate these metrics. Formula (3) is the PSO objective function.

### 5 PSO Routing Simulation Results and Conclusions

We have inserted PSO algorithm codes into NS2 (Network Simulator 2) for WSN routing. In our simulation scenario, 32 sensor nodes move randomly in a 100\*100 meter area at the speed of 0.05 meter/second. To reduce the problem, we assume that all nodes have the same radio transmission power and with the same traffic rate. During 500 seconds simulation, 8 source nodes send data packets at constant bit rate traffics respectively (rate traffics are between 100kbps and 1500kbps) to other 5 destination nodes. Transmission range for all nodes is 20 meter.

QoS evaluation metrics (3) are used to select optimal routing path from source node to destination node. Routing path optimized by PSO, called QoS-PSO, is compared with the performances of QoS-AODV [7] at different data traffic rates. These QoS metrics are BWER and packet delay.

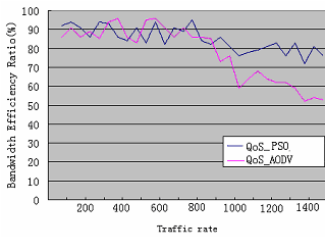


Fig. 1. Bandwidth efficiency ratio

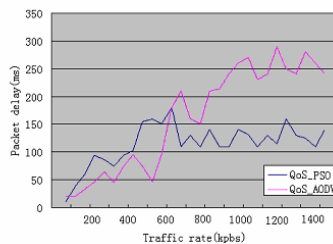


Fig. 2. Packet delay

As shown in Figure 1, when the total amount of traffic sent to the network increases, BWER for QoS-AODV and QoS-PSO keeps its performance and remains higher at traffic rate lower than 800kbps. However, as traffic rate higher than 800kbps, in QoS-AODV the BWER decreases observably due to link congestions and packet drops. When a congestion of a link affects on the level of demanded QoS level, QoS enabled

QoS-AODV pauses communication and prevents any packet drop until another path with enough bandwidth resources. As demonstrated in Figure 1, *BWER* in QoS-PSO only drops about 10%, which shows that a well routing path get less effecting.

Figure 2 shows that packet delay in QoS-AODV is less than QoS-PSO under about 600kpbs traffic rate, but at 600kpbs higher traffic rate, the packet delay in QoS-AODV is longer than that of QoS-PSO about 120ms. It is true that at lower traffic rate, The routing path selected by QoS-PSO is not satisfy due to other QoS metrics are mentioned. Otherwise at higher traffic rate packet delay is mostly lower than that of QoS-AODV because the path selected by QoS-PSO has higher bandwidth efficiency and therefore avoids congestion.

It is not omitted that above two metric improvements are due to the power energy reserve guarantee in routing path selection.

In this paper, we present a QoS based routine algorithm with particle swarm optimal (PSO). The particle swarm behavior in the above approach ensures that WSN maintain network routing, which have an optimal QoS level. As compared to QoS-AODV approach, the particle swarm optimal based routing approach will achieve better QoS performance for WSN application.

## References

1. P, Mohapatra. J Li. C Gui.: QoS in Sensor Ad hoc Networks. Special Issue of IEEE Wireless Communications Magazine. June 3 (2003) 44-52
2. Chen Xhu. M Scott Corson.: QoS routing for sensor adhoc networks. Proceeding of IEEE Infocom, IEEE, 8 (2002) 543- 589.
3. Kennedy J. Eberhart R.: Particle Swarm Optimization. Proceeding of IEEE International Conference on Neural Networks, IEEE, (1995) 1942-1948.
4. Saida Ziane. Abdelhamid Mellouk.: A Swarm Intelligent Scheme for Routing in Mobile Ad Hoc Networks. Proceedings of the 2005 Systems Communications(2005) 2-6
5. Liu, J. Lee, E .A.: Timed Multitasking for Real-Time Embedded Software: IEEE Control System Magazine, February(2003)65-75
6. Chen, B. Jamieson, K, Span.: An Energy-efficient Coordination Algorithm for Topology Maintenance in AdHoc Wireless Networks. MOBICOM2001,July2001
7. C, E, Perkins. E, M, Royer. S, R, Das.: Quality of Service for Ad hoc On-Demand Distance Vector Routing. IETF Internet Draft, work in progress, July 2000.
8. Liang, Zhao. Qilian, Liang.: Distributed and Energy Efficient Self-Organization for Wireless Sensor Networks Wireless Information Networks, Vol. 12, No. 1, January (2005)3-9

# A Novel Multi-agent Automated Negotiation Model Based on Associated Intent

Weijin Jiang<sup>1</sup> and Yusheng Xu<sup>2</sup>

<sup>1</sup> School of computer, Hunan University of Technology,  
Zhuzhou 412008, P.R. China  
jwjnudt@163.com

<sup>2</sup> College of Applied Electronics, Beijing University of Science & Technology,  
Beijing 100081, P.R. China  
yshxu520@163.com

**Abstract.** With the information explosion speeds up the increasing of computing complexity rapidly, the traditional centralized computing patterns are under great pressure to process those large-scale distributed information. However, the agent-based computation and high-level interaction protocols foster the modern computation and distributed information processing successfully. The multi-agent system (MAS) plays an important role in the analysis of the human-interaction theory and model building. This study focuses on the formal description of MAS, the conflict-resolving mechanisms and the negotiation in MAS. The communication between agents has some special requirements. One of them is asynchronous communication. Used communication sequence process (CSP) to describe a model of agents communication with shared buffer channel. The essence of this model is very suitable for the multi-agents communication, so it is a base for our next step job. Based on the communication model, explored the distributed tasks dealing method among joint intention agents and with description of relation between tasks we give a figure of agents' organization. Agents communicate with each other in this kind of organization. The semantics of agent communication is another emphasis in this paper. With the detailed description of agents' communication process, given a general agent automated negotiation protocol based on speech act theory in MAS, then we use CSP to verify this protocol has properties of safety and liveness, so prove it is logic right. At last a frame of this protocol's realization was given.

## 1 Introduction

The theory of Multi-Agent Automated Negotiation involves extensive applying fields and many kinds of methods. The theory mainly lies in Argument Based Automated Negotiation, Game Theoretic Models and Heuristic Approaches. In application, it can be divided into two categories, Agent's Negotiation within MAS and Self-interested between different MAS. Those theories supporting the interior collaboration of MAS are like Self-interested, Joint Intentions and Shared Plans, no matter which are have differences, they have been working under the premise of identical intention and target of Agent within MAS. This text will discuss the Joint Intentions in Multi-Agent Automated Negotiation of MAS[1-4].

If Multi-Agent in MAS interacts successfully, there must be three conditions demanded to be satisfied as below:

- 1) Communication Structure, that is, how to dispatch and take over information between Agent;
- 2) Communication Language, that is, Agent is required to understand the signification of the information;
- 3) Interaction Rules, that is, how to organize the conversation between Agent.

Regarding to the research of Agent Communication Structure, we have proposed TTMAS communication model in the previous parts. In the second section, it will be stressed to analyze Agent’s asynchronous communication mechanism[5,6]. As to the research of Agent Communication Language, presently there have been many abroad, like KQML, FIPA, ACL, Agent Talk, etc., so the language is not the emphasis in our text. Then, research of Interaction Rules is the second emphasis in the text. In the third part, the text will set forth the agreement of Agent Automated Negotiation and its validation. In the forth part, it illustrates and analyzes the complete frame of Agent Automated Negotiation. The fifth is the conclusion of the text.

## 2 Agent Communication Mechanism Analyses

**Definition 1.** Agent is a status course which can accomplish the task automatically with the ability and agreement of communication, for example,  $P_A$  represents the course of Agent A.

**Definition 2.** The course of Agent make the Agent’s ability which can be marked as  $Ability_{P_A}$  and  $TASK_{P_A}$  means to be able to fulfill the task.

The moving status of the static Agent in MAS can be classified as Active, Wait and Run. Agent in the Wait status will be activated after receiving the requests from other Agent and then run. Agent in Run status will negotiate with other Agent or provide services according to the Try-best principle.  $State_{outer}$  stands for the Run status of Agent:

$$State_{outer} ::= Wait \mid Active \mid Run$$

Agent’s collaborating course observed from the outer MAS is the process that Agent runs in the  $I_{outer} = State_{outer}^*$ .

**Definition 3.** Contain the protocol system extremely locking the state, including STOP process in its CSP expression formula.

**Definition 4.** Contain alive protocol system that lock, its CSP expression formula will certainly include part exported to have pass ring of returning.

**Theorem 1.** In an Agent’s collaborating process with Safety and Liveness, the circulation of  $Wait \rightarrow Active \rightarrow Run \rightarrow Wait$  in  $I_{outer}$  will appear at least once to Agent’s launch and acceptance.

Attestation: Obviously, in the circulation of  $Wait \rightarrow Active \rightarrow Run \rightarrow Wait$ , if any one part of Agent can not fulfill the circulation, it means something happened unexpectedly

cause the deadlock or livelock to the system during the collaborating process, so the theorem attested.

**Definition 5.** Buffer channel C is such an Agent which set independent state switch and message buffer to all its relevant Agents and transmit messages for these Agents.

### 3 MAS Interior Agent Cooperation Model

#### 3.1 Agent Cooperating Principle

When Multi-Agent in MAS begins cooperation, for the reason that there is a conform joint intension between Agent, the process of Multi-Agent in MAS works according to the principal of "From each according to his ability, abide by the law and behave oneself", that is, each Agent is trying its best to cooperated with other Agent[7,8].

The cooperation between Agents is aimed at fulfilling a certain tasks. Because tasks can be divided into different but related sub tasks, the tasks from Agent’s point of view can be described as following: a material task can be regarded as sub-tasks’ assembling depending on different ability of Agent in MAS. Combining divided-task-oriented Agent in compliance with sub tasks will be in position to form a furcation tree of  $k(k \geq 2)$ . Relation between sub tasks is relation with or to time sequence. Agent’s organizing relation is determined by the relation between tasks. Description of sub tasks as below:

(1) The sequential relationship of the tasks ( $<$ ), manifests that Agent B’s task can not be begun before fulfilling Agent A’s task. Formalization to be described below:

$$TASK_{P_A} < TASK_{P_B} \models P_A ; P_B$$

Thereinto :  $TASK_{P_A}$  and  $TASK_{P_B}$  respectively means the start-up tenor  $P_A$  and  $P_B$  of Agent A and Agent B are used to fulfill tasks.

(2) The relation of “AND” between tasks ( $\vee$ ), indicates that Agent A and Agent B perform simultaneously sub task  $P_A$  and  $P_B$ . After completing the sub tasks, Agent C begins their common and subsequential task  $P_C$ . Formalization described as below:

$$TASK_{P_A} \vee TASK_{P_B} \models (P_A \parallel P_B) < TASK_{P_C} \models (P_A \parallel P_B) < P_C$$

(3) The relation of “OR” between tasks ( $\wedge$ ), indicates that Agent A and Agent B with the relation of “OR” perform simultaneously sub task  $P_A$  and  $P_B$ , no matter which is fulfilled first, Agent C can begin its subsequential task  $P_C$ . Formalization described as below:

$$TASK_{P_A} \wedge TASK_{P_B} \models (P_A < TASK_{P_C} = \parallel (P_B < TASK_{P_C} = \models (P_A < P_C = \parallel (P_B < P_C =$$

### 3.2 Automatic Negotiation in Agent Protocol

Agent automatic negotiation is the main method for multi-Agent to negotiate, which focus on three aspects lieing in negotiation protocol, negotiation object and negotiation policy. Negotiation protocol and negotiation object act as the textual points, but the negotiation policy is clamping how to look for in Agent each from of negotiation space best in order to reach consistence, concretion content visible literature cited.

Present hypotheses 1 to ensure negotiation agent could each other have partner faith in against due to MAS interior Agent according to Try-Best principle proceed synergic, furthermore MAS possess concurrent combine intent.

**Hypotheses 1.** Negotiation Agent knows each other in negotiation policy.

Be on the negotiation with the result that decision agent toward inter network communication negotiatory condition of Agent automatic negotiatory course mission due to specific assignment require different communication quality guarantee AND specific network insurance. Text take mission negotiation AND inter network communication negotiation as agent automatism negotiation in process two phase.

**Definition 6.** MAS interior agent automatic negotiation course could include two phases. The first phase is based on multi-Agent automatic negotiation whose negotiation object includes task starting time, task ending time and the relation of the tasks; The second phase is the negotiation of Agent’s communicating conditions whose negotiation object include corresponding security policy and network service quality (QoS) .

According to the top analysis talks about with the correlative language behavior academic theories, we say the Agent automatic negotiation correspondence in the procedure to state row word certain for: request, promise, refuse, advise, counter advise. In view of agreement presence overtime event and agent unsolicited message transmission, so increase overtime (timeout) status and inform (inform) state row word that. Communication protocol engine of the communication process state as follows of the agent :

$State_{inner} ::= Started | Requested | Accepted | Refused | Promised | Informed | Advised | CAAd-vised | Timeout | Stopped$

See Fig. 1: Agent automatic negotiation protocol can be divided into information transmission layer, buffer channel layer and Agent negotiation protocol layer from bottom to top, of which buffer channel layer C is one of the needed layers between Agents to realize asynchronous communication. If it will realize point-to-point synchronous communication between Agents, it can do communication directly through channel C. As to the description of Agent automatic negotiation, it mostly focus on Agent negotiation protocol layer, while for the other layers, it only describes their services and running environment in brief. In essence, the function of Agent negotiation protocol layer is the description of process.

The service provided by each protocol layer:

- a. Information transmission layer: being in position to transmit information data between Agents in sequential way and correctly;
- b. Buffering channel C0 and C1 layer: providing Agent automatic negotiation layer with the services described in 2;
- c. Agent automatic negotiation protocol layer : supplying Agent with credibility, efficient negotiation control and policy.

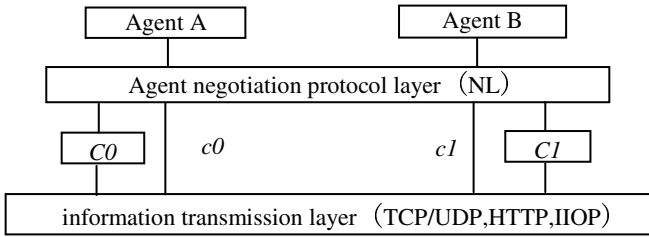


Fig. 1. Agent Automatic Negotiation Protocol Model

Description of Agent negotiation protocol layer functions:

Fig. 2 receive agreement on state vicissitude chart, from the view of agent negotiation starter Among them: arrowhead direction are the flow direction of Agent information; The Recv means a certain message in roger in right connecting; send stand for exactness forward to some information ; Following behind the Recv/ Send is a message type, both state vicissitude picture with state refused and accepted implication negotiatory amphi-with the result that; negotiation the rough and smooth , along with it show that negotiatory terminal status.

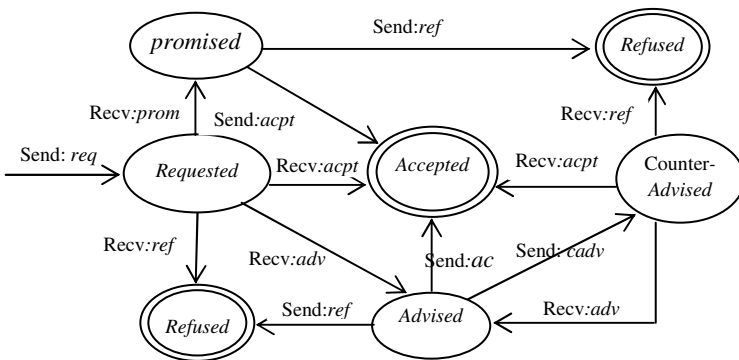


Fig. 2. Agent Negotiation Protocol Statement Vicissitude Figure

The Agent A describe with Agent B whole negotiation procedural not formal as follows: Agent A first of all dispatch negotiation beg of Agent B received solicit aback, toward request message proceed analyses, could as per three strain scene



dispose to : the first thing, in the event of Agent B receivability the solicit of Agent A, those Agent B to Agent A dispatch take send, else dispatch thumb advise, down through upon, the service request block mode, of the such negotiation scene as conventional C / S. the second thing, Provide some Agent B can provide serve of instruct, but because of the restrict of the resource of system can't very much the serve, so the Agent B can put forward to Agent A the serve promises, the Agent A handles Agent B the commitment of serve can proceed very much: Reject or accept. the third thing, The Agent B thinks after analyzing the Agent A request Agent A some items modification within request empess, can satisfy the Agent A request still, like this Agent B after proceeding Agent A some items within request to modification, conduct and actions the suggestion sends out to the Agent A. Agent A for the suggestion of Agent B can operation proceeding as follows: Accept, reject and put forward the counterproposal. either that of toward Agent A counterproposal, Agent B receivability, reject or set own the other one proposal for.

## 4 Conclusions

This text provides a common and communication-based Agent cooperation mode by studying mutual behavior of Agent cooperation. The text also uses some effective format ways to depict automatic negotiation protocol of Agent process and verify the validity of the protocol's logic. Finally, the text makes an implementation frame for this agreement. While using blackboard mode to realize buffer channel in this implementation frame, it provides a deployed agreement stack extra and at last it presents performance analysis and expandable analysis. In addition, as to negotiation between Agent in MAS, because the advantage difference of Agent group negotiating with Agent which has a conform joint intension has great differences on negotiation principle and strategy, the self-interested Agent's negotiation agreement between MAS is our next work under research.

## References

1. Jennings N R ,Faratin P, Lomuscio A R et al.: Automated negotiation: prospects. Methods and challenges[C]. Pacific Rim International Conference on Artificial Intelligence, (2000)
2. Grosz B, Sidner C.: Plans for discourse[A]. In: P. Cohen, Morgan J, Pollack M. eds. Intentions in communication [M]. Bradford Books, MIT Press, (1990)
3. Wang Bin . Zhang Yao-xue, Chen Song-qiao: A communication method of MAS based on blackboard architecture[J]. Mini-Micro Systems, 23(11), (2002) 1355-1358
4. In G. Agha and F.: Decindio, editors, Concur-rent Object-Oriented Programming and Petri Nets, Lecture notes in Computer Science[M]. Springer-Verlag, Berlin, (1998)
5. Jiang Weijin: Modeling and Application of Complex Diagnosis Distributed Intelligence Based on MAS. Journal of Nanjing University(Natural Science), 40(4) ,(2004) 483-496
6. Jiang Weijin: Research on Diagnosis Model Distributed Intelligence and Key Technique Based on MAS. Journal of Control Theory & Applications, 20(6), (2004) 231-236
7. Jiao Wen-pin, Shi Zhong-Zhi.: Modeling dynamic architectures for multi-agent system[J]. Chinese Journal of Computers, 23(7),( 2000) 732-737
8. Mao Xin-jun: Anon-terminating active computing model in multi-agent systems[J]. Journal of Computer Research & Development, 36(7) , (2003) 769-775

# Research on Design and Implementation of Adaptive Physics Game Agent for 3D Physics Game\*

Jonghwa Choi, Dongkyoo Shin, and Dongil Shin\*\*

Department of Computer Science and Engineering, Sejong University,  
98 Kunja-Dong Kwangjin-Gu, Seoul, Korea  
jhchoi@gce.sejong.ac.kr, shindk@sejong.ac.kr,  
dshin@sejong.ac.kr

**Abstract.** In 3D game contents, the physics engine takes charge of the role to increase the reality of game contents. Objects that act by a decided scenario cause a motion problem for the objects as well as for the reality of the objects. To solve these problems, a physics engine was presented. We proposed a design and implementation of the physics game maker based a physics engine. We also showed the architecture and demo simulation of the physics game maker. This paper explains the basic data structures that compose the physics game maker. We explained the main component that is working in the physics game maker, and showed demo simulation through car simulation. The current physics game maker only processes a rigid body, and supports one world.

**Keywords:** Physics Engine, 3D Game Engine, Game Simulation.

## 1 Introduction

In spite of excellent expression of graphics, high-end games need additional physical effects for enhancement of the game's reality [1]. Progress in 3D graphics technology expresses only more graphics aspects in the game, but it does not add enough processing for a realistic reaction of object in game. A physics engine was presented to solve these problems. The physics laws in a game were applied to improve the game's reality in a primitive racing game or arcade game, and are applied in various FPS games through development of hardware. Study of the physics engine has made great progress in three directions. Motion of an object in a game is calculated by various integral methods. Thus, the first direction is a study to improve performance of motion calculation of an object by selection of an effective integral method. Second, is research about the effective calculation method of collision detection, and third is the study of soft body collision in 3D game. This paper explains the structure of the physics game maker and its function, and presents the function of rigid body kinetics in the physics game maker. The study of the performance level of the physics engine has contributed much to our understanding of the efficient structure of the engine itself. Gottschalk and Lin presented a data structure and an algorithm for efficient and exact interference detection amongst complex models undergoing rigid

---

\* The study was supported by a grant of the Seoul R&BD Program.

\*\* Corresponding author.

motion [2]. An understanding of the interaction of multiple parts in the working of the physiotherapy engine was developed from the study of collision detection; one such study led to *voxel's* efficient structure to improve the speed of collision detection [3]. Research that compared the performance of collision detection led to an algorithm (a boxtree), which is defined by the structure of the object [4]. ODE (Open Dynamics Engine) is a well-known open source based physics engine [5]. Math Engine [6], Havok [7] and Meqon [8] are also widely used as commercial physics engines.

## 2 Structure of the Physics Game Maker

Figure 1 shows a general function to implement a physics engine. DS is the data structure for the physics engine, and it defines information of object. SF is the function that calculates object's motion, and it includes the collision system. The utility system manages memory, configuration error, and etc.

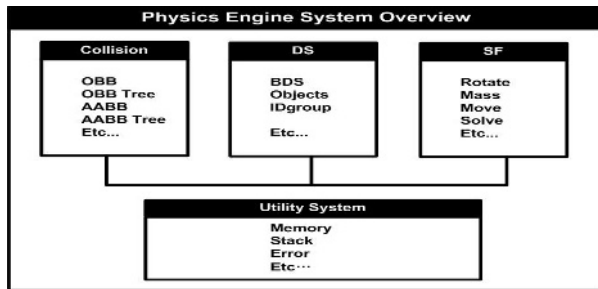


Fig. 1. Basic Functions of Physics Engine

The physics game maker that is proposed in this paper was implemented by including the functions of physics engine in figure 1. Figure 2 shows the integration structure of the physics component and the rendering engine in our physics game maker.

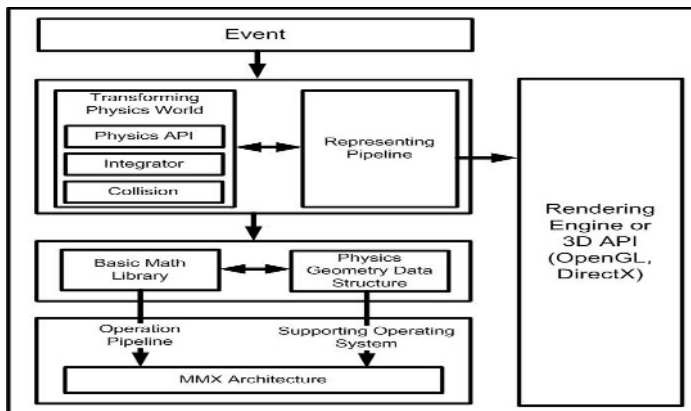


Fig. 2. Integration Structure of Physics Components and Rendering Engine

It is also the system architecture for rigid objects to support effective physics phenomena in 3D games. The physics game maker was implemented by this structure. Our physics game maker was designed mainly for rigid body processing. The physics components define basic polygon as the unit of an object, and polygons have their attributes (mass, position, rotation, etc), and the physics components make a model of the rigid body through association of polygons. Also, the joint relations of polygons were designed to be established according to the properties of the rigid body. Collision detection in world space is executed sequentially through collision detection of inside object polygons and outside object polygons. The physics game maker defines the joint relations between polygons, and controls the motion of the polygons. If the car moves, the physics game maker has to manage the joint relations and the motion of the car. Our physics game maker offers joint relation of four types. To support calculation of the object’s motion, the physics game maker use integral methods of two types (Euler method and Runge-kutta method). Figure 3 shows architecture of the physics game maker in this paper. The physics game maker manages attributes of all game character as the object tree of XML. So, it effectively controls character management and change of game scene. Figure 3 shows structure of the integration object control manager that supports effective character management by XML tree, presents the game physics components that supports physics motion calculation of all object.

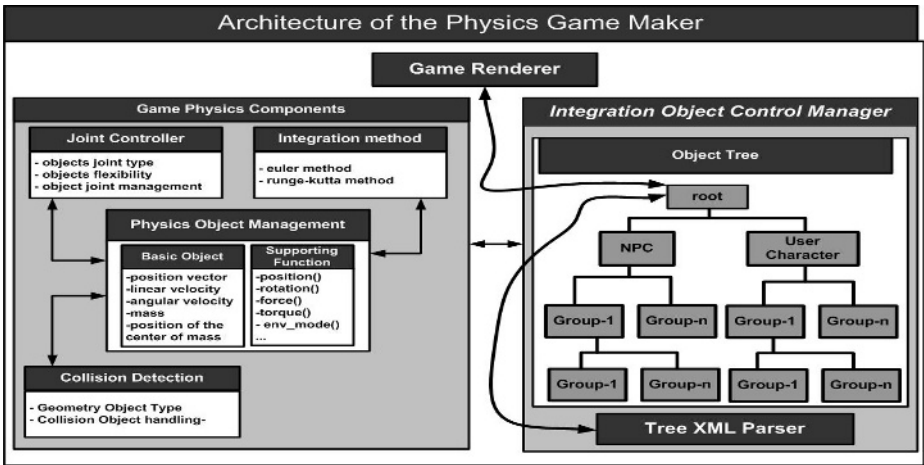


Fig. 3. Integration Structure of Physics Components and Rendering Engine

The game physics components includes four modules, each module offers basic API for all game object. The physics object management includes physical attribute’s value of object and method that calculates changed object’s motion. The physics object management interacts with three modules (joint controller, collision detection and integration method). The collision detection calculates collision response each objects. The integration method takes charge of calculation. The physics game maker supports two integral methods (euler method and runge-kutta method). Figure 4 shows the class diagram of the physic game maker. XML parser creates the

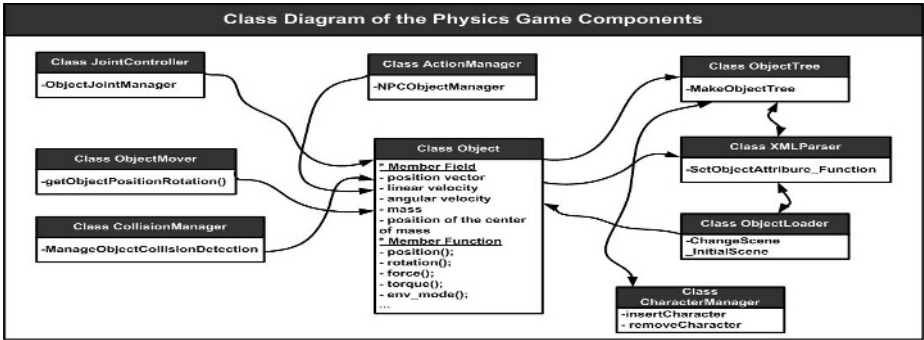


Fig. 4. Class diagram of the physics game maker

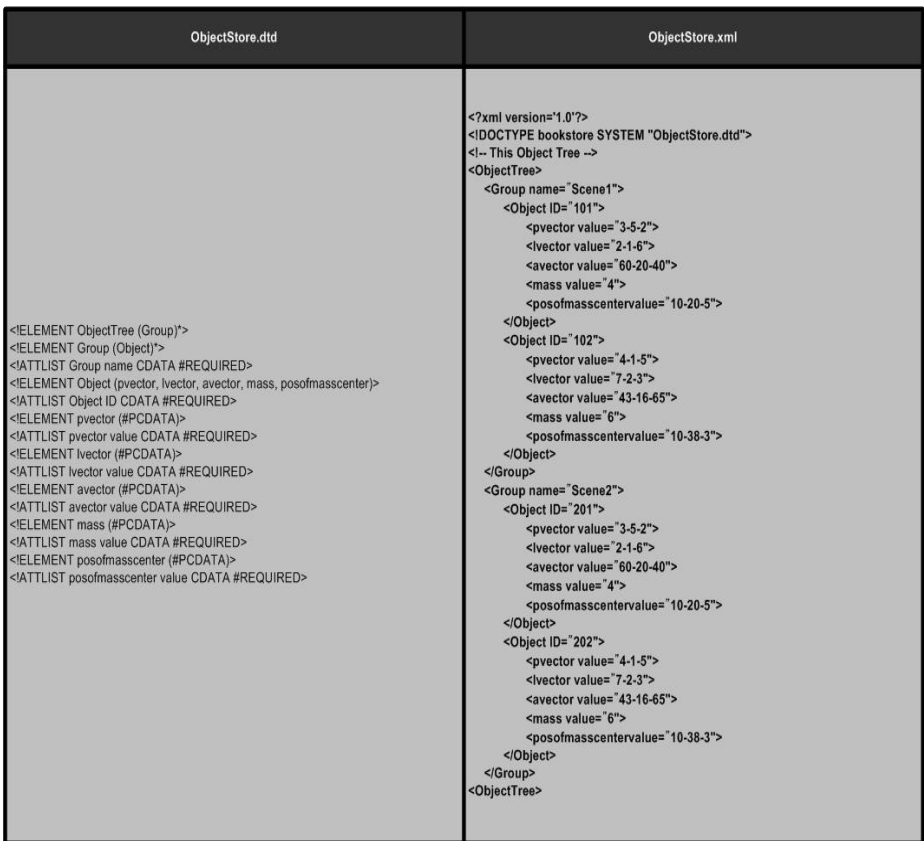


Fig. 5. XML format of the ObjectTree

ObjectTree that consists of all object, the ObjectTree manages each object by object pointer that is included Object node. Figure 5 shows XML format of the ObjectTree. The ObjectStore.dtd defines all sub-elements that consist of the ObjectTree. The ObjectTree includes the Group, the Group includes the Object.

Figure 6 presents a flowchart by character control. If character move in game, the ObjectTree finds a character that moves in the game world, and calculates attribute value of moved object (position, rotation value, etc). If it is stored previous game scene, the physics game maker loads object tree that is stored in database to initialize game scene. The ObjectTree consists of all object’s attribute through XML parse. The initialization of scene executes two processes.

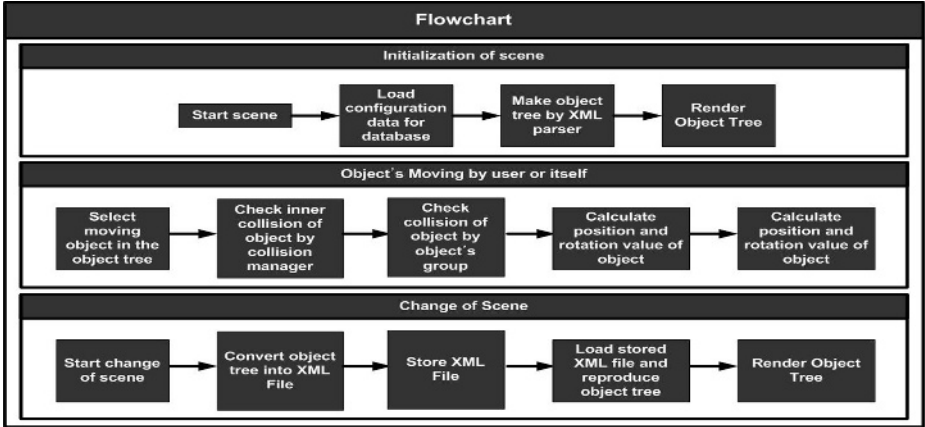


Fig. 6. Flowchart by character control

### 3 Implementation and Evaluations

We present a demo simulation of the physics game maker through car simulation based joint and collision detection. The car simulation that is presented in this section, is applied JointHinge as the Polygon’s joint type.

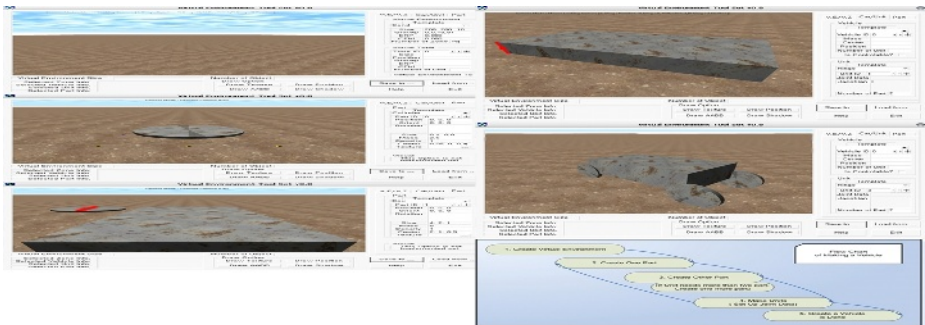
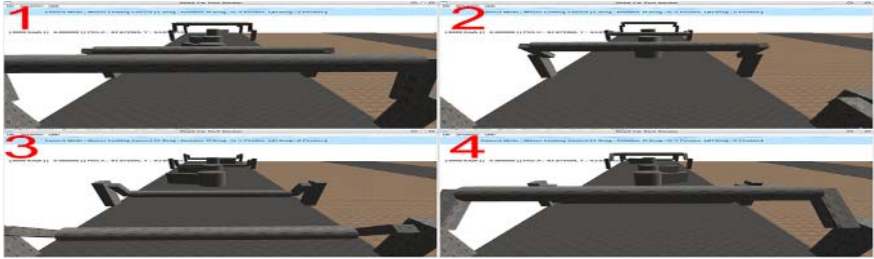


Fig. 7. Object Creation through the ObjectTree Creator base XML data format

Figure 7 show a vehicle’s creation process based on XML data structure, using the physics editor. First, the developer has to establish the virtual environment that is used for all objects. Figure 8 shows performance of joint in the physics engine. We

presented car simulation that is created by the physics game maker. The physics game maker supports scalable object management based XML data structure. Our simulation shows physics master character (car) and physics obstacle. All characters and obstacles is included in the ObjectTree, the ObjectTree can control physical management of all objects.



**Fig. 8.** Screen shot of the Car Simulation

## 4 Conclusions

In 3D game contents, the physics engine takes charge of the role to increase the reality of game contents. Objects that act by a decided scenario cause a motion problem for the objects as well as for the reality of the objects. To solve these problems, a physics engine was presented. We proposed a design and implementation of the physics game maker based a physics engine. We also showed the architecture and demo simulation of the physics game maker. This paper explains the basic data structures that compose the physics components. We explained the main component that is working in the physics game maker, and showed demo simulation through car simulation. The current physics game maker only processes a rigid body, and supports one world.

## References

1. Serviss, B. Seligmann, D.D.: Escaping the world: high and low resolution in gameing. Multimedia. IEEE, Volume 12. Issue 4. (2005) 4-8
2. S, Gottschalk and M,C, Lin and Manocha.: Hierarchical Structure for Rapid Interference Detection. Computer Graphics. Vol 30. (1996) 171-180
3. Lawlor, O, S. Kalee, L, V.: A voxel-based parallel collision detection algorithm. Proceedings of the 6th international conference on Supercomputing, (2002) 285-293
4. Zachmann, G.: Minimal Hierarchical Collision Detection. Proceedings of the ACM symposium on Virtual reality software and technology. (2002)
5. Open Dynamics Engine.: <http://ode.org>
6. Math Engine.: <http://www.mathengine.com>
7. Havok.: <http://www.havok.com>
8. Meqon.: <http://www.meqon.com>

# An Improved TTS Model and Algorithm for Web Voice Browser

Rikun Liao, Yuefeng Ji, and Hui Li

College of Telecommunication, Beijing University of Posts and Telecommunications,  
P.O. Box 128, 100876, Beijing, China  
lyfen@sina.com, {jyf, lihui}@bupt.edu.cn  
<http://oilab.ste.bupt.cn>

**Abstract.** The paper describes a Web voice browser based on improved text-to-speech algorithm and architecture, which making Internet content available by voice. A visual and audible web browser was discussed in terms of HTML files to be tuned with TTS and speech recognition processes. The voice evaluation results show that the system has better voice quality and data identifiability than other voice browsers.

## 1 Introduction

Currently, users obtain services from Internet that offers visual presentation capabilities. For example, a PC with a Web browser that requests and receives HTML documents produced by a Web server. Generally speaking, the current Web browser can only be read and not meet specific requirements.

However, voice browser can provide people a new and easily accessible network which making content available by voice. Therefore, the Web voice browser which combines web and speech synthesis technology has been applied to practice.

The main target of improved LPC algorithm and TTS architecture is to transfer voice data stream into more fluent voice output. With the method, the Web voice browser produces more fluent speech and ensures a naturally sounding system. The system structure also improves the intelligibility and naturalness of the speech compared with other voice browser.

Evaluation test was carried out to evaluate the performance of different voice browser methods. The results indicated the improved TTS voice browser has better performance than other Web voice browser model in the areas of voice quality and data identifiability.

## 2 Web Voice Browser Architecture

The Web voice browser includes three main components s which are HTML text extraction, Chinese syncopation and Chinese speech synthesis. These components will identify HTML text from Web server and pronounce ersatz voice in the output, as were shown in figure 1.



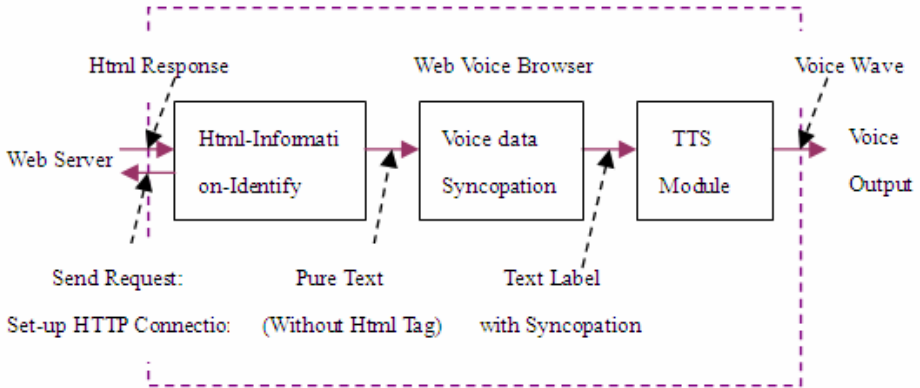


Fig. 1. Main components of Web voice browser

The main target of TTS is to transfer any voice stream appear in computer into fluent voice output. The system is composed of text analysis module, prosodic module and improved TTS speech synthesis module, which was given as figure 2.

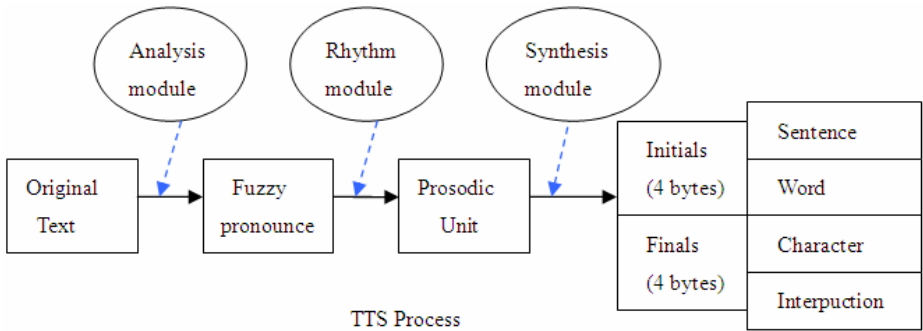


Fig. 2. The process of Chinese Text-To-Speech

Original text processed by analysis module will tell the computer what and how to pronunciation. But this pronunciation mode is still abstract. Rhythm module will fulfill the specific tone and present prosodic characters such as tone, mood, pausal mode and so on. With the result of prosodic modeling, the speech synthesis module will extract the corresponding elementary unit from initialized library, adjust the rhythm characteristic and finally pronounce the accordant sound. The elementary unit can be sentence, words, characters and Chinese initials / finals.

### 3 Improved LPC-Based TTS Algorithm

LPC (Linear Predictive Coding) is an important coding method to a Chinese TTS system because it will influence the prosodic model for speech synthesis. Improved

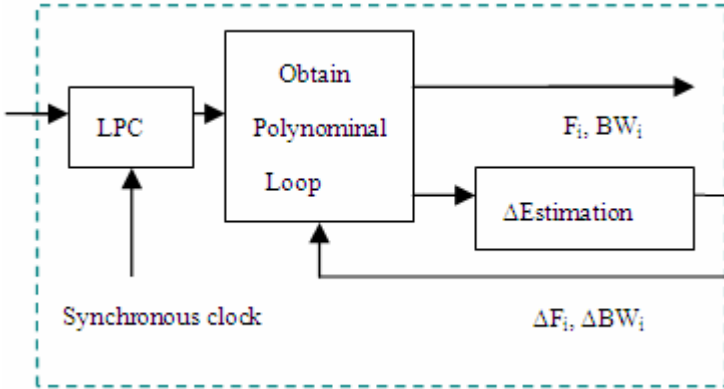


Fig. 3. The improved LPC algorithm for TTS model

LPC is an optimized scheme compared with LPC one. In figure 3, it was used for unit selection and for ensuring smooth formants across the speech synthesis.

The synchronous speech signals for each frame are obtained from the multi-pole model, the previous vector  $s(t)$  in the adaptive LPC buffer must be updated by the coefficients of the newly decoded frame  $s(t-1)$ , the  $t$  and  $t-1$  are the current and previous frame indexes,  $\alpha_\tau$  is the additive coefficient, then the predict coefficient  $p_t$  will be:

$$\Delta p_t = \sum_t [s(t) - \sum_\tau \alpha_\tau s(t-\tau)]^2, \Delta = p_t - p_{t-1}. \tag{1}$$

$F_i$  and  $F_j$  respectively describe the pole frequency and signal sampling frequency. The complex polynomial of multi-pole model can be defined as:

$$z_i = e^{-2\pi(BW_i/F_i) \pm j2\pi(F_i/F_j)}. \tag{2}$$

$F_i$  and  $BW_i$  are respectively the centre frequency and bandwidth. The first derivatives, i.e. temporary dynamics, of formant frequency and bandwidth could be modeled as:

$$p_j(k) = 2^{\frac{j}{2}} F(k). \tag{3}$$

The most likely adjacent poles are at time  $t$  and  $t-1$ . The  $k$  pole has the smallest Euclidean distance with  $i$  pole:

$$k = \arg \text{Min}_k [F_i(t), BW_i(t)] - [F_k(t-1), BW_k(t-1)]. \tag{4}$$

The feature vector for each frame will be  $[F_i, BW_i, \Delta F_i, \Delta BW_i]$  feature vector which used for analysis and synthesis model for each frame of speech.

## 4 Evaluation and Comparison of Speech Quality

The Web voice browser has several different realized methods such as RRN TTS-based system, VoiceXML(Voice eXtensible Markup Language) browser, MEPL (Mixed Excitation Linear Prediction)-based model and so on. RRN-based algorithm model can preferable detect prosodic phrase for Chinese TTS system while VoiceXML browser is an application of XML on voice browser.

In the speech synthesis process, TTS systems may be suffered from unnatural quality for several reasons including the lack of natural prosody, discontinuity around unit joining points and the annoying quality of unvoiced regions. Therefore, the evaluation tests were subsequently carried out to evaluate the performance of the different methods, the MOS(Mean Opinion Score, 5-the best,0-the worst) test and PESQ (Perceptual evaluation of speech quality) were carried out. In the performance comparison, the HTML texts in the database contains 3017 sentences with 32391 syllables, 4195 prosodic words, which are all Internet news selected from a large database to cover a variety of subjects. The results indicated that RRN and VoiceXML were both lower scores than improved LPC TTS-based model.

In the improved LPC TTS model, the synthesized speech signals sounds closer to original speech with the prosody information sufficiently preserved.

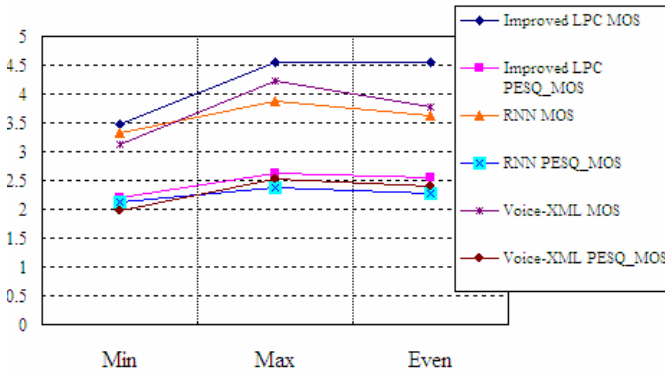


Fig. 4. Evaluation value comparison of different voice browser

Further analysis of the Chinese characters identifiability results indicated that the lower scores for RNN and Voice-XML. The score is the number of correct word pairs divided by the total number of word pairs. The correct rate of the improved LPC TTS in this experiment is about 90.6% ( $29,343/32,390=90.59\%$ ). The precision rate of RNN and VoiceXML are approximate 82.5% ( $26,731/32,390=82.53\%$ ) and 86.3% ( $27,942/32,390=86.27\%$ ) respectively.

As was shown in figure 5, the evaluation outcomes further show that the improved architecture and algorithm is practicable and credible.

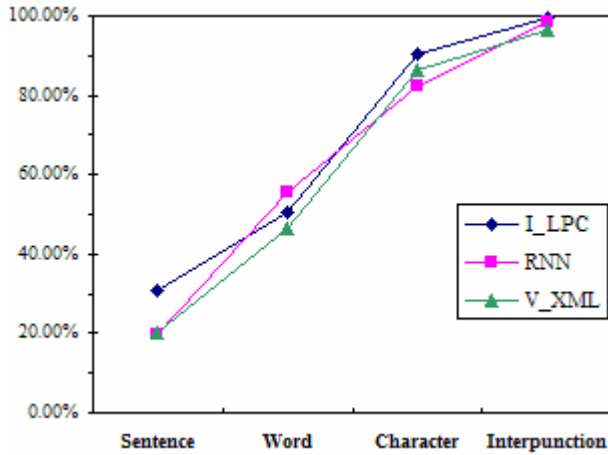


Fig. 5. Identifiability value comparison

## 5 Conclusion

The paper describes an optimized architecture and algorithm for TTS Web voice browser. The Web voice browser is a visual and audible web browser which can improve the precision information for network users.

This improved LPC algorithm helps establish the prosodic model for the Chinese voice browser. The evaluation scores have proved the model can greatly improve the fluency of the synthesized speech and get higher degree of voice naturalness.

## Acknowledgment

This research was jointly supported by the National Science Fund for Distinguished Young Scholars (No. 60325104), National Natural Science Foundation of China (No. 60572021), the SRFDP of MOE (No.20040013001) and Bupt-Hitachi Joint-lab foundation. Thanks for the great help.

## References

1. Chou, F.C., Tseng, C.Y., Chen, K.J. and Lee, L.S.: A Chinese text-to-speech system based on part-of-speech analysis, prosodic modeling, and nonuniform units. in Proc. Int. Conf. Acoustics, Speech, Signal Processing, 1997, pp. 923-926
2. Dutoit, T. : An Introduction to Text-to-Speech Synthesis. Norwell, MA: Kluwer, 1997
3. Liang, S.F., So, A.W., Lin, C. : Model-based synthesis of plucked string instruments by using a class of scattering recurrent networks. IEEE Trans. Neural Networks, vol. 11, no. 1, pp. 171-185, 2000

4. Bao, H., Wang, A., Lu, S. : A Study of Evaluation Method for Synthetic Mandarin Speech. Proceedings of ISCSLP 2002, The Third International Symposium on Chinese Spoken Language Processing, pp. 383-386
5. Chen, W., Lin, F., Li, J. and Zhang, B. : Generation of Chinese Prosodic Phrasing Rules by an Extension Matrix Algorithm. Proceedings of IEEE ICASSP 2002, pp. 489-492
6. Lu, H. M. : An Implementation and Analysis of Mandarin Speech Synthesis Technologies. M. S. Thesis, Institute of Communication Engineering, National Chiao-Tung University, June 2002
7. Yu, M.S., Huang, F.L. : Disambiguating the Senses of Non-Text Symbols for Mandarin TTS Systems with a Three-Layer Classifier. Speech Communication, Vol. 39, Issue 3-4, 2003, pp. 191-229
8. Yan, Q., Vaseghi, S. : Analysis, Modelling and Synthesis of Formants of British, American and Australian Accents. ICASSP, Vol. 1, pp. 712-715, 2003
9. Torajlic, E., Rentzos, D., Vaseghi, S., Ho C.H. : Evaluation of Methods for Parametric Formant Transformation in Voice Conversion. ICASSP, Vol. I, pp. 724-727, 2003
10. Wouters, J., Macon, M.W. : Spectral Modification for Concatenative Speech Synthesis. ICASSP- pp II,941-II.944, 2000

# Network-Based Face Recognition System Using Multiple Images

Jong-Min Kim, Hwan-Seok Yang, and Woong-Ki Lee

Computer Science and Statistic Graduate School, Chosun University, Korea  
mrjyoung@chosun.ac.kr

**Abstract.** The purpose of this study was to propose the real time face recognition system using multiple image sequences for network users. The algorithm used in this study aimed to optimize the overall time required for recognition process by reducing transmission delay and image processing by image compression and minification. At the same time, this study proposed a method that can improve recognition performance of the system by exploring the correlation between image compression and size and recognition capability of the face recognition system. The performance of the system and algorithm proposed in this study were evaluated through testing.

## 1 Introduction

The rapidly growing information technology has fueled the development in multimedia technique. However, demand for techniques involving searching multimedia data in a large scale database efficiently and promptly is still high. Among physical characteristics, face image is used as one of the reliable means of identifying individuals. Face recognition system has a wide range of applications such as face-based access control system, security system and system automation based on computer vision. Face recognition system can be applied to a large number of databases but requires a large amount of calculations. There are three different methods used for face recognition: template matching approach, statistical classification approach and neural network approach[1]. Elastic template matching, LDA and PCA based on statistical classification approach are widely used for face recognition[2, 3]. Among these methods, statistical classification-based methods that require a small amount of calculations are most commonly used for face recognition. Given the proven feasibility of PCA as face recognition method, this study used PCA along with Kenel-based PCA[4, 5] and 2D-PCA[6]. The real-time face recognition system proposed in this study will be available in a network environment such as Internet.

## 2 Network-Based Face Recognition System

Based on the assumption that multiple variations of the face improves recognition accuracy of face recognition system, multiple image sequences were used. To reduce transmission delay, the images were compressed and minimized in the proposed system Fig 1.

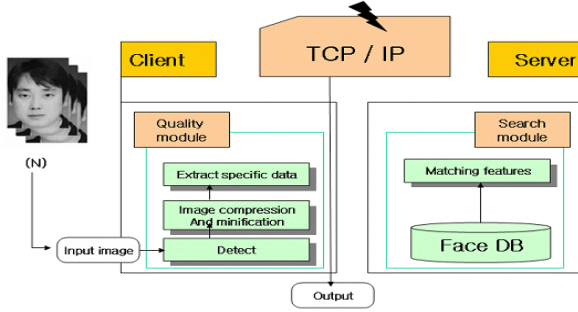


Fig. 1. Composition of the Proposed Face Recognition System

### 3 Face Recognition Algorithms

The real-time recognition accuracy was evaluated using PCA, KPCA and 2DPCA-based algorithms.

#### 3.1 PCA(Principal Component Analysis)

The PCA-based face recognition algorithm calculates basis vectors of covariance matrix ( $C$ ) of images in the following equation.

$$C = \frac{1}{M} \sum_{i=1}^M (X_i - m)(X_i - m)^T \quad (1)$$

Where  $x_i$  represents 1D vector converted from the  $i$  th image in a sequence of images in the size of  $m \times n$ .  $m$  indicates average of total  $M$  images of training face. Maximum number of eigenvectors ( $m \times n$ ) of covariance matrix ( $C$ ) of images are also calculated. Top  $K$  number of eigenvectors are selected according to descending eigenvalues and defined as basisvector ( $U$ )[7]. Feature vectors( $w$ ) of input image( $x$ ) are distributed as basis vectors in the vector space according to the following equation (2):

$$w = U^T(x - m) \quad (2)$$

#### 3.2 2DPCA

While covariance matrix is computed from 1D images converted from input images for PCA, covariance matrix ( $G$ ) is computed from 2D images and the average image for 2DPCA in the following equation (3) [6].

$$C = \frac{1}{M} \sum_{i=1}^M (A_i - E(A))(A_i - E(A))^T \quad (3)$$

Compared with covariance matrix used for PCA analysis, the covariance matrix derived from input images for 2DPCA analysis is smaller. This means that 2DPCA has the advantage of requiring less learning time [6].

### 3.3 KPCA(Kernel Principal Component Analysis)

KPCA face recognition algorithm involves converting input data on a face image into an image using nonlinear functions  $\Phi$ . The converted images are reproduced as eigenvectors of the covariance matrix calculated for a set of nonlinear functions  $\Phi$  and coefficients obtained during this process are used for face recognition in KPCA analysis. In the equation (4), nonlinear function  $\Phi(x)$  is substituted for input image  $x$ , and  $F$  was substituted for the feature space  $R^N$ .

$$\Phi : R^N \rightarrow F, x_k \rightarrow \Phi(x_k) \tag{4}$$

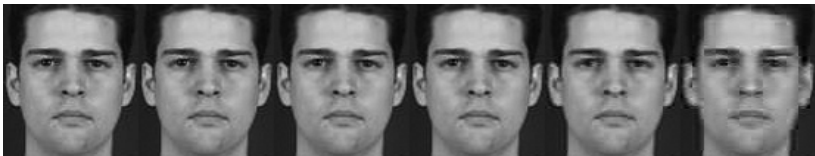
The training matrix and covariance matrix of images in the nonlinear space are presented in the equations (5). The nonlinear function  $\tilde{\Phi}$  in the equation (5) must be mean zero by meeting the requirements for normalization.

$$C^\Phi = \frac{1}{l} \sum_{k=1}^l \tilde{\Phi}(x_k) \tilde{\Phi}(x_k)^T, \quad \sum_{k=1}^l \tilde{\Phi}(x_k) = 0 \tag{5}$$

## 4 Face Recognition Rate

### 4.1 Changes in Recognition Rates with Image Compression

As presented in Fig 2, data file size was reduced but subjective image quality deteriorated as quantization parameters of compressed images increased. As a result, the recognition performance of the system is expected to decline.



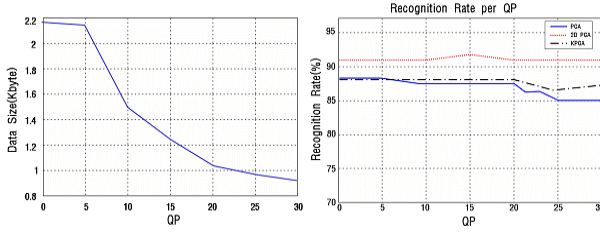
(a) Original image (b) QP=10 (c) QP=15 (d) QP=20 (e) QP=25 (f) QP=30

**Fig. 2.** Changes in visual image with QP value

### 4.2 Changes in Recognition Rates with Image Size

The size of image has an impact on transmission time and computational complexity during the recognition process. Images were passed through a filtering stage to get the





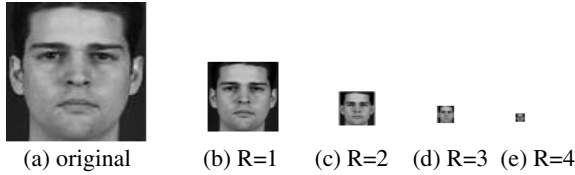
(a) Changes in data size with QP value (b) Changes in recognition rate with QP value

**Fig. 3.** Effects of image compression on data size and recognition performance

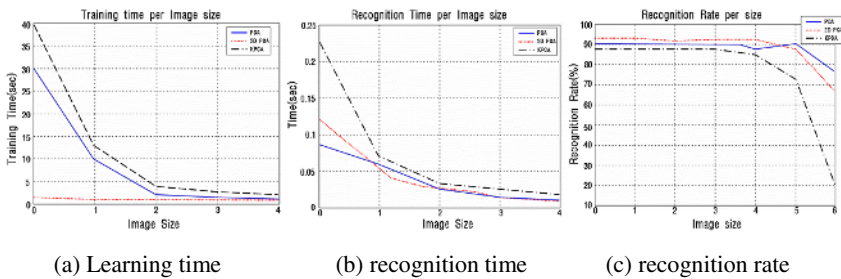
low-low band using wavelet transform. Image size ( $S_R$ ) is defined in the following equation:

$$S_R = \frac{S_{origin}}{4^{(R)}} \tag{7}$$

For instance, the original image is reduced to 25% of its actual size when R equals to 1. Effects of image filtering are presented in Fig 4.



**Fig. 4.** Effects of image filtering



(a) Learning time (b) recognition time (c) recognition rate

**Fig. 5.** Effects of image size on time required for learning and recognizing images and recognition rate

Effects of image size on time required for learning and recognizing images and recognition performance are presented in Fig 5. As shown in Fig 5 (a) and (b), the time required for learning and recognizing images drastically fell as the size of the image was reduced. The recognition rate also dropped when R was less than 4 but stopped its decline and remained almost unchanged when R was 4 or above. This is due to the

fact that image size reduction involves reducing the number of faces in original images and the size of coefficient vectors.

### 4.3 Majority-Making-Decision Rule

The study proposes a face recognition algorithm capable of improving recognition performance of the system. The algorithm is designed to calculate a recognition rate based on the majority, composition and decision-make rules when multiple input images are used. A theoretical estimation of recognition rate ( $P_m$ ) can be calculated in the following equation on the condition that more than half of transmitted images were matched with image models.

$$P_m = \sum_{k=\lfloor n/2 \rfloor}^n \binom{n}{k} P_s^k (1 - P_s)^{n-k} \quad (8)$$

Where  $n$  is the number of images forwarded,  $P_s$  is the average probability of face recognition  $\binom{n}{k}$  is possible number when  $k$  number of features are recognized among  $n$  number of features.  $\lfloor x \rfloor$  is a fixed number that is smaller than  $x$  but the largest value. For instance, when  $P_s$  is 0.94 and three images are forwarded, the value of  $P_m$  is 0.99 based on the majority, composition and decision-making rules.

The proposed algorithm was tested with 3, 5 and 7 images in PCA-, KPCA- and 2DPCA-based real-time face recognition systems. According to the equation (8), the saturation estimation was achieved when  $P_s$  was roughly larger than 0.7 and  $n$  equaled to 5. Five images were therefore used for the test of the proposed system. Test results are presented in Fig 6.

## 5 Experiment Results and Conclusions

The composition of the proposed system is presented in Fig 1. For the test, Chosun-DB (50 class, 12 images in the size of 60\*120) and Yaile DB were used. The test was performed in a network environment, and the optimized value of quantization parameters was applied. Experimental results are presented in Table 1. The performance of real-time face recognition system is measured by the length of time required for learning and recognizing face images, total amount of data transmitted and recognition rate. The amount of data transmitted was reduced to 3610 bytes from 19200 bytes, leading to 81% reduction in transmission delay.

This study proposed a real-time face recognition system that can be accessed across the network. The test of the proposed system demonstrated that image filtering and image compression algorithms reduced transmission delay and the time required for learning and recognizing images without hindering recognition accuracy of the system. This study used multiple input images in order to improve the recognition performance of the system, and the proposed real-time face recognition system

proved robust on the network. Although the system was based on PCA algorithms, it can be integrated with other face recognition algorithms for real-time detection and recognition of face images.

**Table 1.** Comparison of performance between proposed and existing systems

Algorithm	Recognition Rate(%)	Training Time	Recognition Time(sec)
PCA	88.7	28	1.5
Proposed system(PCA)	92.0	16	1.0
2D PCA	91.3	27	0.5
Proposed system(2D PCA )	95.4	6	0.38
KPCA	79.0	2.4(hour)	36
Proposed system(KPCA)	93.5	0.33(hour)	5

## References

1. A.K.jain,R.W.Duin,and J.Mao,"Statistical pattern recognition : a review,"IEEE Trans.on Pattern Analysis and Machine Intelligence,Vol.22,No.1, pp.4-37,Jan.2000.
2. W. Yambor, "Analysis of PCA based and Fisher Discriminant-based Image Recognition Algorithms," Technical Report CS-00-103, Computer Science Department Colorado State University, 2000. 7.
3. Hiroshi Murase and Shree K, Nayar, "Visual Learning and Recognition 3-Dobject from appearance", international journal of Computer Vision, Vol,14,1995.
4. Y. Zhang, C. Liu, "Face recognition using kernel principal component analysis and genetic algorithms",Proceedings of the 2002 12th IEEE Workshop on Neural Networks for Signal Processing, pp.337-343
5. J. Yang, D. Zhang, A. F. Frangi, and J. Y. Yang, "Two-Dimensional PCA : A New Approach to Appearance-Based Face Representation and Recognition, "IEEE Trans. Pattern Analysis and Machine Intelligence, vol.26, no.1, January, 2004..
6. F. Bourel, C.C Chibelushi and A.A Low, "Robust facial expression recognition using a state-based model of spatially localised facial dynamics", Proceedings of Fifth IEEE
7. International Conference on Automatic Face and Gesture Recognition, pp.106-111, 2002.
8. A. S. Georghiades, P. N. Belhumeur and D. J. Kriegman, "From Few to Many : Illumination Cone Models for Face Recognition under Variable Lighting and Pose," IEEE Trans. Pattern Analysis and Machine Intelligence, vol.23, no.6, pp.643-660, June, 2001.
9. Hwan-Seok Yang, Jong-Min Kim, and Seoung-Kyu Park, "Three Dimensional Gesture Recognition Using Modified Matching Algorithm", Lecture Notes in Computer Science LNCS3611 pp224-233, 2005 9. Hwan-Seok Yang, Jong-Min Kim, and Seoung-Kyu Park,

# Reusable Component Oriented Agents: A New Architecture

W.H. Boshoff and E.M. Ehlers

Academy of Information Technology, University of Johannesburg, P.O. Box 524,  
Auckland Park, 2006, South Africa  
willem.boshoff@ey.com,  
eme@rau.ac.za

**Abstract.** This research paper will outline the new Plug-in Oriented Agent Architecture (POAA) and describe the agents that make use of the POAA architecture. POAA agents make heavy use of functional and controller based plug-ins in order to extend the functionality and behavior of the agent. The architecture was designed to facilitate machine learning and agent mobility techniques. POAA agents are created by mounting newly created dynamic plug-in components into the static structure of the agent. The use of plug-ins will greatly improve the effectiveness of researchers, as only a single, standard architecture will exist. Researchers only need to design and develop the plug-in required for their specific agent to function as desired. This will also facilitate the comparison of various tools and methods, as only the components being reviewed need to be interchanged to measure system performance.

**Keywords:** Agent architecture, mobile agent, learning agent, agent plug-in.

## 1 Introduction

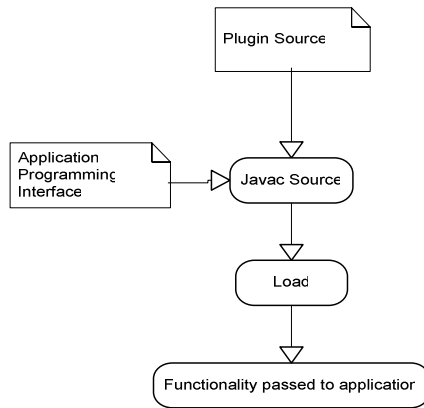
There are numerous implementations of artificial intelligence (AI) agents present in the global computing environment. New types and classifications of agents are introduced on an almost daily basis. There is a multitude of applications for agents. Agents are also used extensively in the expert system environment [1]. Plug-ins are widely used within the computer industry. These are mainly used to add additional functionality to an already existing, static application. Plug-ins can also be used to fix security vulnerabilities within an existing application.

This research paper will aim to find an effective solution to a number of difficulties faced by modern agent researchers by combining the advantages of current agent technologies and modern plug-in theories and technologies.

## 2 Plug-In Theory

A large number of obstacles facing modern agent technology research can be solved by providing a standardized plug-in architecture for researchers. The POAA architecture has been designed to provide a standardized architecture to be used by

researchers that wish to research and develop new agent technologies. Plug-ins are code components written in an extension language that can be loaded at run-time or compile-time, so that an application’s functionality can be extended dynamically. Controller plug-ins can also be used to alter the overall behavior and characteristics of the agent. No agent architecture currently exists that makes use of both functional and controller plug-ins to alter the behavior of the agent. A plug-in architecture comprises of a standardised interface, the application programming interface (API) for the plug-ins, compilation manager, plug-in docking station, plug-in services registry and the plug-in itself [2]. Plug-ins extends a host application at predefined point.



**Fig. 1.** The compilation and loading of plug-ins within an application. The plug-in source code is required to comply with the application API. The source code is then compiled and loaded into the agent’s memory space.

*Definition:* Plug-in – A hardware or software module that adds a specific feature or service to a larger system. The new component simply plugs in to the existing system in order to provide increased functionality [3].

Figure 1 illustrates how the plug-in source code is compiled by a SUN Java compiler. The plug-in, as illustrated in figure 1, is required to comply with the host application’s API. The compiled plug-in is then loaded into the memory space of the host application. The plug-in will therefore expand the functionality of the host application, and allows the host application to call on the functions of the new plug-in. Plug-ins therefore provides a great opportunity and method for easily extending the functionality of an application.

### 3 Plug-In Oriented Agent Architecture

A number of current agent architectures make use of a single plug-in that provides some additional level of functionality to the agent architecture. Some of the agents make use of plug-ins in order to control the agent. For an example of such an agent, please refer to the RatCog architecture [4]. Some agents make use of plug-ins in order

to enhance their own functionality or to make use of different technologies interchangeably. An example of such an agent is the Cougaar agent architecture [5]. Some agents make use of interchangeable plug-ins in order to facilitate different inter agent communication protocols. However, none of these agents make use of all these types of components. Only making use of a single type of plug-in will result in the agents not being flexible, as each type of agent can only be used and configured for a particular task or be used in a particular, predefined environment. None of the agent architectures currently available make use of both agent controller based plug-ins and functionality based plug-ins. Controller based plug-ins refer to those plug-ins that alter the agent's decision structures and behavior. Functionality based plug-ins refer to all plug-ins that extend the agents current abilities.

### 3.1 Breakdown of Individual Components

Based on the discussion above, the Plug-in Oriented Agent Architecture (POAA) is proposed. POAA consists of a number of unique components that interact with each other in predefined protocols. Static components provide the backbone and basic infrastructure of the POAA. The dynamic components can be configured individually by the agent designer. The dynamic components conform to a static interface or template. Interfaces allow a researcher to develop his own components provided that they conform to the component template. The dynamic components are able to interact with the static, infrastructural components; however, they are not able to make changes to the static components. Figure 2 is a complete diagram of the proposed architecture.

#### 3.1.1 Static Components

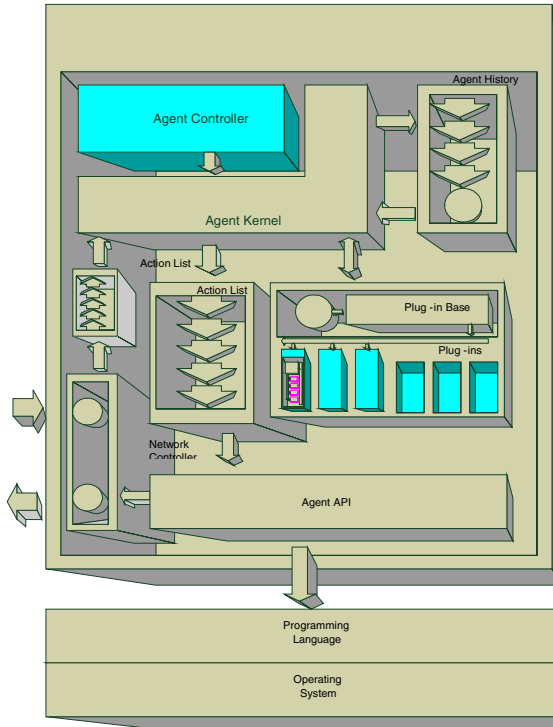
The majority of components within the proposed architecture are classified as static components. This does not refer to their internal state, as a number of these components do have a dynamic internal state. A static component refers to those internal agent components that have a static role and position within the internal structure of the agent. One such example is the action lists that operate as list structures and are used to pass messages between various other components. The lists are classified as static agent components, but they have a dynamic internal state as the messages are passed along.

#### Agent Kernel

The agent kernel is the core and central component of the proposed agent architecture. It is responsible for the initiation of all other components. It is also used to facilitate communication between the agent controller plug-in and other internal components, such as the agent history and plug-in bank. All migration activities are initiated and controlled by the agent kernel. The kernel will ensure that the entire operational history of the agent is preserved during migration.

#### Agent History

The agent history component fulfils a number of roles within the POAA. The history component will ensure that all actions taken by the agent during its actions as well as each action's individual result will be stored inside the memory module. The agent history can also be used by the researcher to determine if an appropriate quality of



**Fig. 2.** This figure illustrates the proposed agent architecture. The figure also illustrates how the individual components are situated within the agent architecture and how they are interlinked with each other.

service has been provided by the agent. The second function provided by the history component is support for different agent learning capabilities. The implementation of the agent's learning capabilities is done with the support of the WEKA [6] project.

### Plug-In Bank

The plug-in bank is the main repository for all functional plug-ins that are integrated into the agent. It is the responsibility of the plug-in bank component to effectively and efficiently route all command requests from the controller plug-in and agent kernel to the correct functional plug-in and to return the results of the requested operation back to the command requestor.

### Action Lists

The action list component is used to pass messages between different components within the agent architecture. It is a simple container that encapsulates a list type data structure and operates as a first-in-first-out (FIFO) queue. The action list has been configured to send text based messages and does not currently facilitate the passing of non-text message objects. There are two areas where the action list component is implemented. One instance of the action list component is used to transport messages between the agent kernel and the plug-in bank. A second instance of the action list is used to transport messages between the agent kernel and the network controller.

**Agent API**

The agent API component serves as a connection point between the agent kernel and the programming environment/operating system of the agent, as well as between the kernel and the network controller. The agent API allows the agent to perform operating system specific actions on the host server.

**Network Controller**

The network controller is the agent's main interface with the outside world. Through the network controller, the agent is capable of communicating with other agents that are residing on either the same agent environment or on a remotely hosted agent environment. The network controller can also communicate with other service providers or services that are not necessarily agent related. The network controller allows the agent to communicate with external web servers or even remote databases. Such communication is however dependant on the functional plug-in and how it has been configured to operate.

**3.1.2 Dynamic Components**

The Dynamic components are those components that are not provided by the agent environment by default. The dynamic components have to be supplied by the developer or researcher. They also function as the main drivers of the agent. The dynamic components include the following:

**Agent Controller Plug-In**

The agent controller is the primary dynamic component of the agent architecture. Each agent is required to have a single controller. The agent controller plug-in contains the decision structures of the agent. It is responsible for achieving the objectives of the agent. All planning processes are encapsulated by this component. The controller plug-in is slotted into the agent kernel using a plug-in wrapper. The agent controller does not maintain a static structure and is dependant on the developer and his programming requirements.

**Functional Plug-In**

The functional plug-in components provide additional functionality to the agent. It is used to assist the agent controller in achieving its objectives. If a developer wishes to create his own plug-ins, he will be required to implement the template and add the functionality that he requires into the supplied methods. The creation of new functional plug-ins makes it extremely easy for a developer to quickly and effortlessly develop and implement new functionality for an agent. New agents can be created by simply adding new combinations of existing functional plug-ins into the agent's plug-in bank. Each plug-in can support any number of functions. The functionality of each plug-in therefore does not have any limits. It is also possible for individual functions within a plug-in to interact with the network controller.

**4 Conclusion**

In this paper, a new architecture was presented for a generic intelligent agent that can be modified by the individual researcher in order to create a new customized agent. Agent modification can be done by adding or removing individual plug-ins from the



agent architecture. The researcher is also able to create new Controller Plug-ins that allows the researcher to specify new agent behavior models. The proposed architecture makes it easier for a researcher to quickly and effortlessly design and prototype a new type of agent. By making use of a standard internal agent structure, the researcher is capable of comparing different agent technologies within the same agent, thereby making it easier for the researcher to compare the different technologies for efficiency and effectiveness.

## References

1. Tarmel Kennion; 1999; A Community of Expert-System Agents; Technical report from NASA ACE and Department of Computer Science, North Carolina Agricultural & Technical State University, Greensboro, USA.
2. Alf Wang, Anders Hanssen, Bard Nymoen; 2000; Design principles for mobile, multi agent architecture for cooperative software engineering; Norwegian University of Science and Technology, Norway; 4th IASTED International Conference on Software Engineering and Applications, Las Vegas, USA, November 2000
3. Unknown Author; 2005; Ask I.T.: Glossary: O – Z; The University of Queensland, Brisbane Australia; <http://askit.uq.edu.au/glossary/glossaryo-z.html>; 2006-03-26.
4. Christopher G. Prince; John Talton; Istvan S. N. Berkeley and Cengiz Gunay; 2000; RatCog: A GUI maze simulation tool with plugin “rat brains”; University of Minnesota, Duluth, USA; SciP (Society for Computers in Psychology) 2000 Conference, New Orleans, November 16th, 2000
5. Aaron Helsinger, Michael Thome, Todd Wright; 2004; Cougaar: A Scalable, Distributed Multi-Agent Architecture; BBN Technologies, Cambridge, USA; Proceedings from the IASTED Conference on Computational Intelligence, 7/4/2005 - 7/6/2005, Calgary, Alberta, Canada.
6. Ian H. Witten, Eibe Frank; 2000; WEKA, Machine Learning Algorithms in Java; University of Waikato, Hamilton, New Zealand; Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations, Morgan Kaufmann Publishers, 2000, pp. 265-320

# Expected Utility Maximization and Attractiveness Maximization

Ka-man Lam and Ho-fung Leung

Department of Computer Science and Engineering  
The Chinese University of Hong Kong  
{kmlam, lhf}@cse.cuhk.edu.hk

**Abstract.** When a decision-maker needs to choose among a number of choices, each having a certain probability to happen, one of the traditional ways discussed in economics is to calculate the expected utility of each choice and chooses the choice with the maximum expected utility. However, most of the humans do not do so in real situations. One of the famous example is the Allais paradox. The reason why most of the people do not maximize the expected utility is that people have different attitudes towards risk in different situations and people are generally risk-averse. In this paper, we proposed a model of decision-making, which considers the risk attitude of a player, reputations of other players and the payoffs of the choices. We call this model *attractiveness maximization*. We find that the model of attractiveness maximization can be used to explain a situation, the Allais paradox, which violates the expected utility theory.

## 1 Introduction

In economics, decision-making under uncertainty often refers to the problem that a decision-maker chooses among a number of choices, each having a certain probability to happen. According to the Expected Utility Theory [4], a decision-maker makes decision by calculating the expected utility of each choice and chooses the choice with the maximum expected utility. However, most of the humans are not expected utility maximizers in real situations. Very often, people choose the choice which has a lower expected utility.

One of the famous examples showing most of the humans violate the expected utility theory is the Allais paradox [2]. There are two sets of choices. In the first set, choice  $A$  is a 100% chance of receiving 1 million while choice  $B$  is a 10% chance of 5 million, an 89% chance of 1 million, and a 1% chance of nothing. In the second set, choice  $C$  is an 11% chance of 1 million, and an 89% chance of nothing while choice  $D$  is a 10% chance of 5 million, and a 90% chance of nothing. According to the expected utility theory, those who prefer choice  $A$  to  $B$  should prefer choice  $C$  to  $D$ . However, it is found that most people prefer  $A$  to  $B$  and  $D$  to  $C$ , which violate the expected utility theory.

In an experiment conducted by Kahneman and Tversky [3], there are two choices: one pays an utility of 3000 with probability of 1, while the other pays an

utility of 4000 with probability of 0.8 and otherwise nothing. If a decision-maker maximizes the expected utility, the decision-maker should choose the second choice as the expected utility is higher. However, the experiment shows that out of 95 respondents, 80% choose the first choice, while only 20% choose the second one. Kahneman and Tversky suggest that the reason why most of the decision makers violate the expected utility theory is that there is risk in the choices and humans have their own attitudes towards risk. When there is risk in the choices, people are usually risk-averse and prefer getting something for sure.

In the Allais paradox and the experiments conducted by Kahneman and Tversky [3], there is a 100% probability of gain in some choices, while there is a risk of no gains in other choices. In these experiments, most people prefer the outcome with 100% probability of gain rather than taking the risk of no gains to exchange for a higher possible payoff. These people are attracted by the 100% probability of gain, even the expected utilities of the choices are lower. On the other hand, some people are attracted by choices with higher expected utilities and are willing to take the risk of no gains.

Instead of calculating the product of probability and utility, some people consider the amount of gain to be more important than the probability of gain in making decisions, like those who pay money to enter lucky draws. These people are not averse to risk, they are willing to take the risk of losing money to seek for utility. On the other hand, some people consider the probability of gain to be more important than the amount of gain in making decisions, like those who prefer the choices with 100% probability of gain, they are averse to risk. So, we use people's attitudes towards risk as a weight of the probability of gain and the amount of gain to calculate the *attractiveness* of a choice. We call the decision-making model *attractiveness maximization*.

In the next section, we introduce the theory of expected utility maximization. In section 3, we introduce the model of attractiveness maximization. In section 4, we compare the two models. In the last section, we conclude the paper and suggest possible future works.

## 2 Expected Utility Maximization

When there is a number of choices, each decision-maker has his own preference over the choices. This preference can be expressed by a utility function. The more the choice the decision-maker prefers, the higher the returned value of the utility function. So, a decision-maker will choose the choice having the maximum utility. Very often, the choices do not happen for certain. There is a certain probability for the choice to happen. The expected utility theory suggests decision-makers to calculate the expected utility of the choices, which is the product of the utility of the choice and the probability that the choice will happen, and choose the one with the maximum expected utility. In a choice, there may be more than one outcome and each outcome has a certain probability to happen. Then the expected utility of the choice will be the weighted sum obtained by adding the utility values of outcomes multiplied by their respective probabilities.

### 3 Attractiveness Maximization

Formally, a player chooses a choice in which the outcome has the maximum *attractiveness*, which is a real number in  $[-1, 1]$ . The attractiveness of an outcome is the returned value of a decision-making function  $f_d(r, rep, u)$ , where  $r \in [0, 1]$  is the risk attitude of the decision-maker,  $rep \in [0, 1]$  is the probability of gain ( $u \geq 0$ ) and  $u \in [-1, 1]$  is the value of the outcome. A decision-maker having risk attitude equals to 0 is the most risk-averse, while the one having risk attitude equals 1 is the most utility-seeking. Note that the risk attitude of a risk-neutral decision-maker is not necessary 0.5, instead, it is a value which satisfies the condition in axiom 4 below.

In some choices, there are several outcomes. Then we calculate an average of the attractiveness of the outcomes in which  $u \neq 0$  if the outcomes involve gain or loss only. If some outcomes involve gain while some involve loss, we calculate a weighted sum of the average attractiveness of the outcomes with  $u > 0$  and the average attractiveness of the outcomes with  $u < 0$ , using the risk attitude as a weight. This is because utility-seeking players are attracted by the outcome of gain, while risk-averse players are resisted by the outcome of loss.

From Cambridge Dictionaries Online [1],

- *Risk* is “the possibility of something bad happening.”
- *Averse* is “strongly disliking or opposed to.”
- *Seek* is “to try to find or get something.”

Following the definitions from the dictionary, risk-averse means strongly oppose to the possibility of something bad happening. In other words, a person is risk-averse means he prefers getting something for sure or with a high probability, and minimizes the possible loss. According to the dictionary, risk-seeking means to try to find or get the possibility of something bad happening. However, people are actually seeking for gain rather than risk. In other words, a utility-seeking person is willing to take a risk in order to obtain a higher payoff.

The function  $f_d$  must satisfy the following axioms:

1.  $f_d$  is continuous.
2.  $f_d = 0$  for  $u = 0$ .
3. If  $u > 0 > u'$ ,  $f_d(r, rep, u) > f_d(r', rep', u')$  for all  $r, r', rep, rep'$ .
4. If  $u \geq u'$ ,  $f_d(r, rep, u) \geq f_d(r, rep, u')$ .
5. If  $rep \geq rep'$ ,  $f_d(r, rep, u) \geq f_d(r, rep', u)$ .
6. (*Cautiousness of risk-averse agents*) There exists a value  $r_0 \in [0, 1]$  such that
  - (a) for  $u > u' > 0$ ,  $f_d(r, rep', u) < f_d(r, rep, u')$  if and only if  $r < r_0$  and  $rep > rep'$ .
  - (b) for  $0 > u > u'$ ,  $f_d(r, rep', u) > f_d(r, rep, u')$  if and only if  $r < r_0$  and  $rep > rep'$ .
7. (*Adventurousness of utility-seeking agents*) There exists a value  $r'_0 \in [0, 1]$  such that
  - (a) for  $u > u' > 0$ ,  $f_d(r, rep', u) > f_d(r, rep, u')$  if and only if  $r > r'_0$  and  $rep > rep'$ .

- (b) for  $0 > u > u'$ ,  $f_d(r, rep', u) < f_d(r, rep, u')$  if and only if  $r > r'_0$  and  $rep > rep'$ .
- 8. (*Indifference of risk-neutral agents*)  $f_d(r, rep', u) = f_d(r, rep, u')$  if and only if  $r_0 \leq r \leq r'_0$ ,  $rep > rep'$  and  $u > u'$ .

It is obvious that the domains of the inputs of  $f_d$  are continuous, so  $f_d$  should be continuous. Axiom 2 states that the attractiveness of an outcome with no gain or loss is zero. Axiom 3 states that the attractiveness of an outcome with gain must be greater than that of an outcome with loss, for any value of risk attitude and probability of gain. Axiom 4 states that if there are two outcomes with the same probability of gain, but with different payoffs, then the one with higher payoff must be more attractive. Axiom 5 states that if there are two outcomes with the same payoff, but with different probabilities of gain, then the one with higher probability of gain must be more attractive.

Suppose there are two outcomes, both involve gain, one with a higher probability of gain but with a lower payoff, while the other with a higher payoff but with lower probability of gain. Axiom 6a states that the first one is more attractive than the second one for a risk-averse decision-maker because a risk-averse person prefers getting something for sure or with a higher probability. So, a risk-averse decision-maker views a choice with a higher probability of gain to have higher attractiveness, even the gain is lower. On the other hand, a utility-seeking person prefers getting a higher payoff even it is not for sure or with a lower probability, a utility-seeking decision-maker views a choice with a higher payoff to have higher attractiveness, even the probability of gain is lower. This brings about axiom 7a. Axiom 8 states that a risk-neutral person is indifference of the two cases.

Suppose both two outcomes involve loss, one with a higher probability of gain (that is lower probability of loss) but with a lower payoff (that is higher loss), while the other with a higher payoff (that is lower loss) but with lower probability of gain (that is higher probability of loss). Axiom 6b states that the second one is more attractive than the first one for a risk-averse decision-maker because a risk-averse person does not take the risk of losing more. So, a risk-averse decision-maker views a choice with a lower loss to have higher attractiveness, even the probability of loss is higher. On the other hand, a utility-seeking person is willing to take the risk of losing more to exchange for a lower probability of loss. A utility-seeking decision-maker views a choice with a lower probability of loss to have higher attractiveness, even the possible loss is higher. This brings about axiom 7b. Again axiom 8 states that a risk-neutral person is indifference of the two cases.

Below is an example of the function  $f_d$ , satisfying the above axioms:

$$f_d = \begin{cases} (1 - r) \times rep + r \times u & \text{for } u > 0 \\ 0 & \text{for } u = 0 \\ r \times rep + (1 - r) \times u & \text{for } u < 0 \end{cases} \tag{1}$$

### 4 The Allais Paradox

In the Allais Paradox, there are two sets of choices. In the first set,

- *Choice A.* A 100% chance of receiving 1 million.
- *Choice B.* A 10% chance of 5 million, an 89% chance of 1 million, and a 1% chance of nothing.

In the second set,

- *Choice C.* An 11% chance of 1 million, and an 89% chance of nothing.
- *Choice D.* A 10% chance of 5 million, and a 90% chance of nothing.

According to the expected utility theory, if one prefer choice *A* to *B*,

$$u(1) > 0.1u(5) + 0.89u(1) + 0.01u(0).$$

Rearranging the expression gives

$$0.11u(1) > 0.1u(5) + 0.01u(0),$$

and adding  $0.89u(0)$  to each side yields

$$0.11u(1) + 0.89u(0) > 0.1u(5) + 0.9u(0).$$

It follows that choice *C* must be preferred to choice *D*.

However, it is found that most people prefer *A* to *B* and *D* to *C*, which violate the expected utility theory. Let us use attractiveness maximization for decision-making. For choice *A*, we take  $rep = 1$  and  $u = 0.2$ . For choice *B*, there are two outcomes which involve gain, so we take an average of the attractiveness of the two outcomes. In outcome 1, there is a 10% chance of 5 million, we take  $rep_1 = 0.1$  and  $u_1 = 1$ . In outcome 2, there is a 89% chance of 1 million, we take  $rep_2 = 0.89$  and  $u_2 = 0.2$ . For choice *C*, we take  $rep = 0.11$  and  $u = 0.2$ . For choice *D*, we take  $rep = 0.1$  and  $u = 1$ . Using Equation 1, the attractiveness (corrected to 3 d.p.) of the four choices and the selections of the choices for each risk attitude are shown in Figure 1.

Risk attitude	0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1
Choice A	1	0.92	0.84	0.76	0.68	0.6	0.52	0.44	0.36	0.28	0.2
Choice B	0.495	0.506	0.516	0.527	0.537	0.548	0.558	0.569	0.579	0.59	0.6
Choice C	0.11	0.119	0.128	0.137	0.146	0.155	0.164	0.173	0.182	0.191	0.2
Choice D	0.1	0.19	0.28	0.37	0.46	0.55	0.64	0.73	0.82	0.91	1
Selections	A	A	A	A	A	A	B	B	B	B	B
	C	D	D	D	D	D	D	D	D	D	D

Fig. 1. The attractiveness and the selections of the choices for each risk attitude

From the figure, we can see that the most risk-averse player, with risk attitude  $r = 0$ , prefers  $A$  to  $B$  and  $C$  to  $D$ . This is because this player is the most risk-averse and considers only the possibility of gain in making decisions. As the probability of gain in choice  $A$  is 100% and that in choice  $B$  is only 99%, this player prefers  $A$  to  $B$ ; and the probability of gain in choice  $C$  is 11% and that in choice  $D$  is only 10%, this player prefers  $C$  to  $D$ . For utility-seeking players, with risk attitude  $0.6 \leq r \leq 1$ , they prefer  $B$  to  $A$  and  $D$  to  $C$ . As these players are utility-seeking and the possible gain in choice  $B$  and  $D$  is up to 5 million, these players are attracted by the high payoff and prefer  $B$  to  $A$  as well as  $D$  to  $C$ . We can see that these two cases,  $r = 0$  and  $0.6 \leq r \leq 1$ , is the same as suggested by the expected utility theory that those who prefer  $A$  to  $B$  must prefer  $C$  to  $D$ . For risk-averse players, with risk attitude  $0.1 \leq r \leq 0.5$ , they prefer  $A$  to  $B$  and  $D$  to  $C$ . These players are risk-averse but they consider possibility of gain as well as the amount of gain in making decisions. Since the probability of gain in choice  $A$  is higher than that in choice  $B$  and the probability of getting 5 million in choice  $B$  is relatively low, these players prefer  $A$  to  $B$ . In choosing  $C$  or  $D$ , the probability of gain in both cases are very close but the gain in choice  $D$  is relatively much higher than that in choice  $C$ . So, these players prefer  $D$  to  $C$ . This matches the decisions of most humans.

In Allais paradox, most humans violate the expected utility theory. However, using attractiveness maximization can better model human decision-making.

## 5 Conclusion and Future Work

In making decisions under uncertainty, one of the traditional ways discussed in economics is to calculate and maximize the expected utility. However, most of the humans do not do so in real situations. This is because humans have different risk attitudes in different situations and humans are usually risk-averse. To model this, we proposed to calculate and maximize attractiveness and use risk strategies. We found that the model of attractiveness maximization can be used to explain the Allais paradox, which violates the expected utility theory. As future work, we will apply the model of attractiveness maximization and risk strategies to other types of games. Also, we will find out the conditions for the existence of risk strategy equilibrium in different types of games.

## References

1. Cambridge dictionaries online. <http://dictionary.cambridge.org/>.
2. M. Allais. Le comportement de l'homme rationnel devant le risque: 367 critique des postulats et axiomes de l'ecole americaine. *Econometrica*, 21:503–546, 1953.
3. D. Kahneman and A. Tversky. Prospect theory: An analysis of decision under risk. *Econometrica*, 47(2):263–291, 1979.
4. J. von Neumann and O. Morgenstern. *Theory of Games and Economic Behavior*. Princeton University Press, 1944.

# Modeling Negotiation in Combinatorial Auctions Based on Multi-agent

Man-Yin Shi<sup>1</sup> and Shan-Li Hu<sup>1,2</sup>

<sup>1</sup> Department of Computer Science and Technology, Fuzhou University,  
Fuzhou 350002, China  
shimanyin@126.com

<sup>2</sup> Key laboratory of Computer Science, Chinese Academy of Sciences,  
Beijing 100083, China  
husl@fzu.edu.cn

**Abstract.** Combinatorial auctions can be used to reach efficient resource and task allocations in multi-agent system where the items are complementary or substitutable. Determining the winners is NP-complete. This paper presents a negotiation model which is based on the multi-agent, and gives the basic characteristics and the satisfied conditions of winner determination which is determined by the solution of Nash negotiation and indicate the rationality and fairness of this method based on the analysis about the utility increment of the bid agent and the benefit increment of the auction agent.

**Keywords:** Combinatorial auction, multi-agent; utility increment, benefit increment.

## 1 Introduction

Combinatorial auctions, where agents can bid on bundles of items, are popular autonomy-preserving ways of allocating items (goods, tasks, resources, services, etc.). They are relatively efficient both in terms of process and outcome, and are extensively used in a variety of allocation problems in economics and computer science. More recently, there are many researches in this field. Takayuki Ito et al had studied a combinatorial auction protocol among experts and amateurs, and only considered the case of single-skilled experts [1,2,3,4]. Determining the winners in such auctions is a complex optimization problem, but recent research has delivered winner determination algorithms that can optimally solve the problem for quite large numbers of items and bids in practice. An equally important problem, which has received much less attention, is that of bidding. We think that bid agents need to negotiate with the auction agent for some subjective and/or objective reasons: the preference of bidders, the value of items, the capability of items and so on. The auction agent will reset the combination and price of the items after each negotiation until increasing both the benefit of the auction agent and the utility of the bid agent.

This paper represents a new idea that auction agent negotiates with bid agents. In the first round, the bid agents express themselves preference and the auction agent learns the knowledge, then the auction agent presents the primary combination and the price



of each bundle. The bid agent learns and reasons the new combination, then expresses the preference again. The negotiation will repeat like this until both the benefit increment of the auction agent and the utility increment of the bid agent are greater than zero. The auction will fail if they don't come to agreement when the negotiation reaches the last round. This idea comes more in line with the rule of market, because it don't need to combine all the items, which needs exponential time [5]. This model can be used in electric power market, automobile service market, and the auction of pollution authority and so on.

## 2 The Modeling Negotiation

Wooldridge represented that any negotiation frame consists of four parts [6]: the set of a negotiation, the protocol of a negotiation, the strategy of a negotiation, the rule of a negotiation. This paper uses evaluation mechanism to express the rule of the negotiation, makes it as the criterion to determine the agreement, and adds the protocol to the strategy of the negotiation. So the negotiation model consists of three parts: the set of negotiation, the strategy of negotiation, and the evaluation mechanism.

### 2.1 The Set of Negotiation

We define  $M$  as the set of the items needs to be auctioned.

**Definition 1.**  $Ag = \{ a_0, a_1, \dots, a_n \}$  is the negotiating participators which indicates the set of agents participating the negotiation, where,  $a_0$  denotes the auction agent,  $a_1, \dots, a_n$  denote the bid agents . We discuss one negotiates with others, namely,  $n \geq 2$ , auction agent negotiates with all the bid agents.

**Definition 2.**  $W_V = \langle w, v \rangle$  is the vector of the negotiation problem which express the vector of the combinations of the items and their prices, where,  $W_{V_i} = \langle w_i, v_i \rangle$  denotes the vector of the problem that auction agent negotiates with the  $i$ th bid agent, and  $w_i$  indicates the combination of the  $i$ th agent's preference,  $v_i$  is the price they negotiated.

**Definition 3.**  $V^{W_{V_i}} = \langle V^{w_i}, V^{v_i} \rangle$  is the value vector of the negotiate problem, which expresses the value on the negotiate problem vector  $W$  of agent  $a_i$ .

### 2.2 The Evaluation Mechanism of the Negotiation

We assume that  $V_i$  is the psychological price function of the  $i$ th agent, and the bid agent  $a_i$  will bid honestly according to  $V_i$ .  $w_j \subseteq M$  is the set of items, and  $V_{ij}(w_j)$  is its psychological price function value which is the maximum price that agent  $a_i$  bid on  $w_j$ .

It is note worthy that the psychological price function satisfies two characteristics as follows: ①  $V_{ij}(w_j) \leq V_{ij}(w'_j)$ , if  $w_j \subseteq w'_j \subseteq M$ , namely, there are no item has negative price. ②  $V_{ij}(\phi) = 0$ , namely, bid agent will not pay money in the case of getting no good.

The items exchange suiting for combinational auctions has one of the following characteristics at least: complementarity and substitutability.

Suppose that the auction agent divides the items  $M$  into  $m$  disjoint sets  $w_1, \dots, w_m$ , and  $w_1 \cup w_2 \dots \cup w_m = M$ , assume the psychological price function of it is  $u(w_1, w_2, \dots, w_m)$ , and  $u(w_1, w_2, \dots, w_m) = u(w_1) + u(w_2) + \dots + u(w_m)$ .

And the maximum price the bid agent bidding is

$$V(w_1, w_2, \dots, w_m) = \sum_{j=1}^m \max\{V_{ij}(w_j), 1 \leq i \leq n\}$$

It expresses that the auction agent will choose the maximum bid price of each combination  $w_j$  as the eventual winner.

We assume that the negotiation is over now, and in the end the combination of the item has  $k$  sets  $w'_1, \dots, w'_k$  and  $w'_1 \cup w'_2 \dots \cup w'_k \subseteq M$ , and the eventual bid price is:

$$\bar{V}^v = \bar{V}^{v_1} + \bar{V}^{v_2} + \dots + \bar{V}^{v_k}$$

Given the above assumptions, Nash represented that the problem relating to the conflict of interest can be boiled down to a negotiation situation of two parties (here the two parties indicate the auction agent and one of the bid agents). We give a commonly mathematics model (modeling Nash negotiation) of a negotiation situation of two parties based on the utility theory. Nash had proved that the solution of this kind of model satisfies certain principle of rationality and fairness. As a result the method of modeling Nash negotiation has always been thought to be an important method to resolve the problem of multi-party conflict of interest.

We use  $Q_1(w'_1, \dots, w'_k)$  to express the utility increment of the bid agent and  $Q_2(w'_1, \dots, w'_k)$  express the benefit increment of the auction agent.

To the bid agent, after negotiating, the utility of it increment is

$$Q_1(w'_1, \dots, w'_k) = Q_1(w'_1) \times Q_1(w'_2) \times \dots \times Q_1(w'_k)$$

Where

$$Q_1(w'_i) = V_{ii}(w'_i) - \bar{V}^{v_i}$$

If  $Q_1(w'_i) > 0, 1 \leq i \leq k$ , then the benefit of the bid agent  $a_i$  increases after negotiating, otherwise reducing. If  $Q_1 > 0$ , then the benefit of each bid agent increases after negotiating, otherwise at least one of them reduces.

To the auction agent, after negotiating, the benefit increment of it is

$$Q_2(w'_1, \dots, w'_k) = \sum_{j=1}^k \bar{V}^{v_j} - u(w_1, w_2, \dots, w_m)$$

If  $Q_2 > 0$ , then the benefit of the auction agent increases after negotiating, otherwise reducing.

We know that modeling Nash negotiation consists of two basic elements: negotiation space and conflict point of the negotiation. To combinatorial auctions, structuring as follows: to any possible  $(w'_1, \dots, w'_k)$ , ordered pair  $(Q_1, Q_2)$  determines a possible result, so there are:

$$S = \{(a, b) \mid a = Q_1(w'_1, \dots, w'_k), b = Q_2(w'_1, \dots, w'_k), k \in n\}$$

Because the combinatorial auctions carry through negotiation between the auction agent and the bid agent, the basic precondition of the combinatorial auction that can be accepted by both the party after negotiating is they all get certain benefit. Evidently, the partition combination according to  $w_1, w_2, \dots, w_m$  will bring no benefit to both parties, namely,  $Q_1 = 0, Q_2 = 0$ , and then  $d = (0, 0)$  is the negotiation conflict point. The general form of the negotiation model is made up of the dualistic ordered group  $(S, d)$ .

We define the following extreme value problem as the Nash negotiation model. The solution of it is also the solution of Nash negotiation.

$$\max_{\substack{(a, b) \in S \\ (a, b) \geq d}} a \cdot b$$

Therefore the above formula is equal to the extreme value problem:

$$\max \left\{ \left[ \sum_{j=1}^k \max \{V_{ij}(w'_j), 1 \leq i \leq n\} - \sum_{j=1}^k \bar{V}^j \right] \cdot \left[ \sum_{j=1}^k \bar{V}^j - u(w_1, w_2, \dots, w_m) \right] \right\} \tag{1}$$

The solution of this extreme value problem is the combination auction determined in the negotiation sense. Hence, if  $Q_1(w'_i) > 0, 1 \leq i \leq k$  and  $Q_2(w'_1, \dots, w'_k) > 0$ , the negotiation is over.

### 2.3 The Strategy of Negotiation

We assume that the set of negotiation  $Ag = \{a_0, a_1, \dots, a_n\}$  begins at the moment of  $t_0$  and negotiates the negotiation problems vector  $W = \langle w, t \rangle$ , at the moment of  $t_0$ , agent  $a_i, 1 \leq i \leq n$  deliveries suggestions  $V^{W_i} = \langle V^{w_i}, V^{t_i} \rangle$  to agent  $a_0$ , after agent  $a_0$  receiving the suggestions, she learns it, then submits new problem vector  $W' = \langle w', t' \rangle$ . At a certain moment  $t_i$ , the alternate strategies are as follows:

- (1) Agent  $a_i, 1 \leq i \leq n$  evaluates the current negotiation problem through learning, and calculates  $Q_1(w_i)$  Then deliveries new suggestion  $V^{W'_i} = \langle V^{w'_i}, V^{t'_i} \rangle$  to agent  $a_0$ .
- (2) After receiving the suggestion of agent  $a_0$ , she also learns it, and calculates  $Q_2(w_1, \dots, w_k)$ . If  $Q_2(w_1, \dots, w_n) > 0$  and the utility of the problem vector that the other side expresses is within the limit that agent  $a_0$  can receive then the negotiation successes, and the final selling price is  $V_{a_i}^{v_i}, 1 \leq i \leq k$ , otherwise, the negotiation has not reach an agreement, this round negotiation is failing and needs to further negotiate.

- (3) Check whether the negotiation is overtime, namely,  $(t_i - t_0) > t_{\max}$ , where,  $t_{\max}$  is the threshold value of the negotiation which is an advance enactment, the negotiation will stop once it is overtime, otherwise go to the next round.

### 3 The Basic Characters of the Winner Determination Determined by the Solution of Nash Negotiation

Generally the utility function and the benefit function are both have the property of continuous concave, according to this, we can easy to prove that the set  $S = \{(a, b) \mid a = Q_1(w'_1, \dots, w'_k), b = Q_2(w'_1, \dots, w'_k), k \in n\}$  is a convex-compact set according to it. Appropriate combination can bring benefit to both the auction agent and the bid agent, namely, there exists  $(w'_1, \dots, w'_k)$  which makes  $Q_1(w'_i) > 0, 1 \leq i \leq k$  and  $Q_2(w'_1, \dots, w'_k) > 0$ . Thereupon, the negotiation model  $(S, d)$  satisfies the conditions of the Nash theorem [7]. The utility increment of the auction agent and the benefit increment of the bid agent determined by (1) are unique according to the Nash theorem, and this is the agreement reached by the negotiation.

We must make it clear that, to the common function form, the solution of (1) may not be unique, but the utility increment of the auction agent and the benefit increment of the bid agent is the same (or equal to) for different solution. Thereupon, any solution of the extreme value problem can be accepted by both the auction agent and the bid agent, they put up the same preference (utility) when choosing different resolution. But because of the particularity of the real problem, the utility function and benefit function has especial forms which make the solution of the extreme value problem is unique [8].

### 4 The Rationality and Fairness of the Combinatorial Auctions Determined by the Solution of the Nash Negotiation

The solution of the Nash negotiation satisfies a series of axioms so that we can analysis the rationality and fairness of the combinatorial auctions determined by the model of the Nash negotiation.

Because the solution of the Nash negotiation satisfies the individual rationality, the merchandise combination  $(w'_1, \dots, w'_k)$  determined by it satisfies  $Q_1(w'_i) > 0, 1 \leq i \leq k$  and  $Q_2(w'_1, \dots, w'_k) > 0$ . That is, this property reveals a rational condition of an acceptable combination. Because if this condition is destroyed, it must be that one of the two parties cannot accept one of the combinations.

The Pareto optimization of the Nash negotiation solution indicates that no other combination can improve the benefit of both parties that is got by the combination determined by the extreme value problem. This property reflects the rational action and rationality of both parties when they choose the combination making the agreement be reached.

The symmetry of the Nash negotiation solution indicates that both of the parties have equal power and dominator, this embodies the fairness and rational choice in market of the combinatorial auctions determined by this method, and destroys to engross the market.

## 5 Conclusion

The combinatorial auctions can not only improve the efficiency of auctions but also reduce the risk of the bidder. We represent a negotiation model based on the multi-agent, and give the basic character and satisfied conditions of determining the winner which is determined by the solution of the Nash negotiation and show the rationality and fairness of the method based on the analysis of the utility increment of the bid agent and the benefit increment of the auction agent. But some work needs advanced research, such as how the agent learn the suggestion submitted by the other side, comparing with others work, some provement, and simulate the auction process using programming, etc.

## Acknowledgements

This paper is supported by the National Natural Science Foundation of China under Grant NO.60373079, NO.60573076, and by the Foundation of the Chinese Academy of Sciences under Grant NO. SYSKF0505.

## References

1. Takayuki Ito, Makoto Yokoo, and Shigeo Matsubara. Towards a Combinatorial Auction Protocol among Experts and Amateurs: The case of Single-Skilled Experts. AAMAS,2003
2. Anton Likhodedov and Tuomas Sandholm. Methods for boosting revenue in combinatorial auctions. AAI, 2004
3. Benoît Hudson and Tuomas Sandholm. Effectiveness of Query Types and Policies for Preference Elicitation in Combinatorial Auctions. AAMAS, 2004.
4. Tuomas Sandholm. Algorithm for optimal winner determination in combinatorial auctions. AI,2002
5. Eyal Kushilevitz. Privacy and Communication Complexity. IEEE, 1989.
6. Michael Wooldridge. An Introduction to Multi-Agent Systems. ISBN 0-471-49691-X. Copyright © 2002, John Wiley & Sons, Inc 26.
7. Roh A E. Axiomatic models of bargaining. Springr Verlag, New York,1979
8. Avriel M. Nonlinear programming. Prenc---hall Co. Inc., 1976.

# A Methodology for Agent Oriented Web Service Engineering

Hongen Lu and Manish Chhabra

Department of Computer Science and Computer Engineering  
La Trobe University  
Bundoora, Melbourne  
VIC 3086, Australia

helu@cs.latrobe.edu.au, M.Chhabra@latrobe.edu.au

**Abstract.** Web Service (WS) has been widely accepted in recent years. It is becoming the next paradigm to deploy business services on the Web. However, building WS is not an easy task due to the complexity of various business processes and different communication protocols. In this paper, we propose a methodology to efficiently develop Web Services by plugging in software agents. This agent oriented method uses software agents as building blocks of WS, and exploits the commonly available agent development tools to accelerate the whole development cycle. Software agents in this methodology not only implement the business processes, but also enrich the functions of WS. Our approach provides a solution for Web Service engineering. A case study in loan application service is presented to show the benefits and advantages of this agent oriented service engineering methodology.

## 1 Introduction

Web Service (WS) has been widely accepted in recent years. It is becoming the next paradigm to deploy business services on the Web. The World Wide Web is evolving from a sea of information to a service oriented. WS is one of the fastest growing areas of information technology. Web Services expose business processes over the Internet and promise more business opportunities by providing a common protocol that can be used by web applications to communicate with each other over the web. Web services are described in XML and are communicated over existing HTTP infrastructure using SOAP (Simple Object Access Protocol). While WS is bringing revolution in the Web Wide Web by making it a more dynamic environment however still there is an increasing need to apprehend its current capability before exploiting its full potential [3]. Building WS is not an easy task due to the complexity of various business processes and different communication protocols. With the help of software agents (SA) wrapped with WS, the information provided by WS can be made more efficient and more dynamic in nature. Another benefit of wrapping agents into web services is that developers can easily and efficiently reuse widely accepted existing agent based systems as WS. We emphasize on using existing agent development tools which would speed up the development process of WS.

## 2 WAIWS Methodology

To meet the increasing demand of various WSs, an engineering method is much needed to systematically develop reliable WSs. In this paper we present an agent oriented methodology for Web Service engineering, Wrapping Agents into Web Services (WAIWS). In WAIWS method, software agents are not only used for creating, storing, and providing information as they are done by a WS itself, but agents can also be used for continuous monitoring of changes in the information provided by WS and can put together the information gathered from different sources in a more resourceful manner. WAIWS exploits available agent development tools to accelerate the development process using software agents as building blocks.

As [6] suggests that with the current trend in WS, the level of service personalization must be restricted, because of the trade-off between service complexities and abundance of different users requirements. The unsatisfied users must perform additional data processing at user side. This design also fulfils the requirement of customization and service users are able to fit the service into their individual requirements and expectations for the reason that the processing can now be done within the service provider's vicinity. Hence, the wrapping results in more competent Web Services. The following five steps are given to implement our proposed wrapping methodology in JACK [7]:

- 1) Define the adapter interface. This adapter is used to communicate between agent and WS. The method bodies of this adapter interface will be implemented in the agent. They generate the events which in turn invoke the agent's plan associated with these events.
- 2) Define the events that agent will handle. These events are generated by adapter methods for the agent to process. Agent oriented WS is active, it can handle and create events for others to process. This feature of agent makes the whole WS more dynamic and adaptive for any changes in the environment.
- 3) Next is the business process which brings into play when the events are generated. This is the plan of the agent which is associated with the events posted in the environment. Any business process is associated with a plan for agents to carry out. The correspondence of business process and agent plan makes the service flexible and adaptable for any future changes. Without rewriting other parts of the service, developers can simply modify plans to reflect the changes needed.
- 4) Implement the agent and within the agent give the method bodies defined in adapter interface in step 1. In the body part of an agent, plans to execute business processes are given. In BDI model of JACK, agent plans are closely related to business processes. Even more, plans are reusable. Developers can composite or extended current existing plans. This will significantly shorten the development period. Reusing already existed and tested business plans also increase the reliability of the whole services.
- 5) Finally we create the WS and the agents are invoked either when the request is generated to WS or in an asynchronous manner. The calls to agents are made through the adapter.

### 3 A Case Study in Financial Service Domain

In this section we present a case on financial service domain, practically the processing and evaluation of customer loan applications. By deploying the loan processing service online, the banking industry can save millions of dollars on spending of branching and staffing. It will also bring convenience for clients since this WS is available online all the time. Bank customers can access this service any where any time.

#### 3.1 Online Loan Application Service

Loan Application Service (LAS) is an online service for the users to submit their loan applications to the bank. This service defines a method `submitLoanApplication()` which takes all the information of the customer applying for the loan. This method starts the processing for this new application and generates an *event* of new loan application hence invoking the agent Loan Processor Agent. Information required by this service is gathered by the agent, it communicates with other in-house agents to get the final outcome of application. The loan processing events are managed by four agents in the Figure 1.

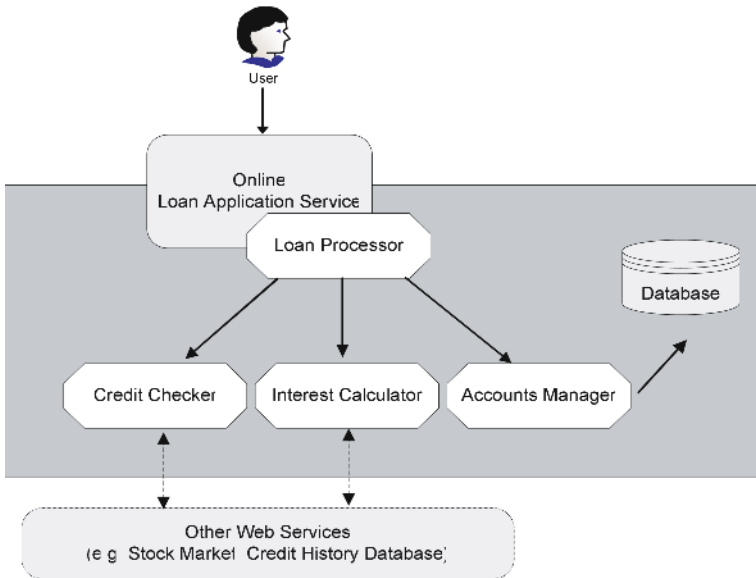


Fig. 1. Online Loan Application Service

This service does all the processing for the submitted application and sends the outcome back the requester. The message format described in WSDL file of this web service gives the description of type of variable it receives and sends back, see Figure 2.



### 3.2 Loan Processor Agent

To achieve this functionality agent Loan Processor (aLP) is wrapped with the service using the methodology described in previous section. The agent with its accustomed capabilities is able to communicate with other in-house agents that provide help to verify credit history of the customer and to find out current interest rates. Since the interest rates may vary depending on the external factors like Federal Reserve bank rate, agents will need to access external services.

```

<wsdl:message name="submitLoanApplicationRequest">
  <wsdl:part name="name" type="xsd:string" />
  <wsdl:part name="details" type="xsd:string" />
  <wsdl:part name="amount" type="xsd:double" />
  <wsdl:part name="customerId" type="xsd:string" />
</wsdl:message>
<wsdl:message name="submitLoanApplicationResponse">
  <wsdl:part
    name="submitLoanApplicationReturn"
    type="xsd:string" />
</wsdl:message>
<wsdl:portType name="LoanApplication">
  <wsdl:operation name="submitLoanApplication"
    parameterOrder="name details amount customerId">
    <wsdl:input
      message="impl:submitLoanApplicationRequest"
      name="submitLoanApplicationRequest" />
    <wsdl:output
      message="impl:submitLoanApplicationResponse"
      name="submitLoanApplicationResponse" />
    </wsdl:operation>
  </wsdl:portType>

```

**Fig. 2.** WSDL of Loan Application Service

The agents form a system to process loan applications. Agent Credit Checker (aCC) checks the credit history of users by communicating with external service. Agent Interest Calculator (aIC) computes current interest rates after retrieving information from other sources like Federal Reserve bank rates. The account Manager Agent (aMA) stores the applicant details in the database if the applicant is not the existing customer. Agent Loan Processor (aLP) communicates with these agents and based on the information obtained the application is accepted or rejected. Using the proposed methodology the whole system can be made online.

The adapter implemented by aLP wraps the agent with online loan application web service. The methods defined by the adapter take the customer details as the

parameter and will post the event for the agent to start processing on this data. The set of actions that need to be performed using this data are defined as the plan of agent. LAS presents the whole system online will define the method to receive all the customer details and pass them to the agent using the adapter.

```

agent LoanProcessorAgent extends Agent
{
    #handles event LoanApplication;
    #uses plan ProcessApplication;
    #posts event LoanApplication la;
    #posts event DoCreditCheck dcc;
    #posts event CalculateInterest ci;
    #posts event CheckAccounts ca;

    CreditCheckAgent(String name)
    {
        super(name);
    }

    void check(//data parameters//)
    {
        postEvent(la.(//data parameters//));
        postEvent(dcc.(//data parameters//));
        postEvent(ci.(//data parameters//));
        postEvent(ca.(//data parameters//));
    }
}

```

Above is a segment of the source code of loan processor agent. This agent posts four events, LoanApplication, DoCreditCheck, CalculateInterest, CheckAccounts, but handles only one event, LoanApplication. The other events are handled by other in-house agents. This *event* of loan processor agent is described as:

```

event LoanApplication extends Event
{
    #posted as check(// data members //)
    {
        // Every event has a plan associated
    }
}

```

### 3.3 Agent Plan

Finally, the *plan* where the business process resides is executed whenever the above event is invoked, the plan is to process the loan applications based on the data members (information about the client). The plans of other SA in the picture also start their processing, like do credit check for the customer, check their credits and debits with the bank and also determine the current interest rates.

```
plan ProcessApplication extends Plan
{
  #handles event LoanApplication la;

  body()
  {
    //Application processing is done here
    //this is the body of plan, and the
    //corresponding plan code is included here.
  }
}
```

## 4 Conclusion

Currently WSs are developed in an ad hoc manner. To meet the increasing demand of WSs, an engineering approach is much needed. In this paper we propose an agent oriented methodology for WS engineering, WAIWS. It exploits available agent development tools to accelerate the development process using software agents as building blocks. Steps are given and described on how to build WSs. WAIWS methodology can churn out more efficient way of providing information through WS, which are capable to respond to frequent changes in the environment. Software agents in this methodology not only implement the business processes, but also enrich the functions of various WSs. Our method provides a solution for Web Service engineering. A case study in loan application service is presented to show the benefits and advantages of this agent oriented service engineering methodology.

## References

- [1] M. Wooldridge and N. R. Jennings. Intelligent Agents: Theory and practice. The Knowledge Engineering Review, 10(2): 115-152, 1995
- [2] Johnson P Thomas, Mathew Thomas and George Ghinea: Modeling of Web Services Flow. Proceedings of the IEEE International Conference on E-Commerce (CEC'03)
- [3] Pradhan, S.; Lu, H.; Using SOAP Messaging to Coordinate Business Transactions in Multi-Agent Financial Web Services; Volume 1, books@ocg.at, iiWAS05, pg. 109-120, September 19-21, 2005
- [4] Web Services Architecture: WWW – page: <http://www.w3.org/TR/ws-arch/>
- [5] Michael Winikoff: Simplifying Agent Concepts. SRI International, AIC, Tuesday 5th June 2001
- [6] J. Rykowski, W. Cellary, Virtual Web Services – Application of Software Agents to Personalization of Web Services, ICEC'04, Sixth International Conference on Electronic Commerce.
- [7] The Agent Oriented Software Group: What is Jack? WWW – page: <http://www.agent-software.com/shared/products/>
- [8] Tim Finin, Jay Weber: Specification of KQML Agent-Communication Language. June 15th, 1993, The DARPA Knowledge Sharing Initiative External Interfaces Working Group.

# Deliberate Soccer Agents Powered by Resource-Bounded Argumentation

Takumi Nisikata<sup>1</sup> and Hajime Sawamura<sup>2</sup>

<sup>1</sup> Graduate School of Science and Technology, Niigata University  
8050, 2-cho, Ikarashi, Niigata, 950-2181 Japan  
`nisikata@cs.ie.niigata-u.ac.jp`

<sup>2</sup> Institute of Natural Science and Technology  
Academic Assembly, Niigata University  
8050 2-cho Ikarashi, Niigata, 950-2181 Japan  
`sawamura@ie.niigata-u.ac.jp`

**Abstract.** The RoboCup Soccer Simulation league is brought to public attention as an intriguing application of agent technology. In this paper, we introduce argumentation mechanism into soccer agents, in order to enhance soccer agents of reactive nature with an ability of deliberation. In doing so, we take into account resource-boundedness and acceleration of argumentation for deliberate but quick decision and action. We demonstrate some advantages of arguing soccer agents with a good balance between reactivity and deliberation with experimental results.

## 1 Introduction

Soccer agents have been designed mainly as reactive ones so far, and soccer games usually proceeds according to the predefined game strategies. Very few attempts have been made at soccer agents who have both capabilities of reactivity and deliberation, although real soccer players explicitly communicate defense and offense information among each other during games in a highly dynamic and uncertain environment. The reason might arise from the official game rules of the RoboCup Soccer Simulation league [1].

We, however, have challenged to build deliberate soccer agents by means of argumentation to which much attention recently has been paid as a promising mechanism for interacting agents [2] [6], aiming at intelligent soccer agents like humans. In this paper, we propose soccer agents with arguing capability as a form of deliberation for communication, collaboration and decision-making. The paper also turns out to yield an evidence to show the effectiveness and usefulness of argumentation in agent studies, which takes into account resource-boundedness for deliberate but quick decision and action.

The paper is roughly organized as follows. In Section 2 to 4, we describe our argument model for soccer agents, deliberate and reactive argumentation protocols and arguing soccer agents architecture. In Section 6 to 8, we introduce some speed-up techniques for accelerating argumentation and agent communication, and knowledge conversion from quantitative to qualitative for building

dynamically changing soccer knowledge. In Section 8, we illustrate two phases of arguments such as who goes for a ball, and formation to be employed in the next game. Finally, we present the effectiveness of resource-bounded but deliberate soccer agents through quantitative game results.

## 2 An Argument Model for Soccer Agents

In this paper we will not go into a formal definition of argumentation framework, due to the space limitation. We just note that the knowledge representation language for argumentation we introduced is a fragment of so-called extended logic programming (ELP) [5] with plausibility for rules, and the win and loss of arguments are decided using the argument superiority criteria such as preference of arguments, evidence and directness of arguments. interested readers should refer to [6] and [3] for the details.

## 3 Two Types of Argumentation for Soccer Agents

As we have seen in the previous section, argumentation proceeds with mutually casting arguments and counter-arguments built from each agent's knowledge base. Argumentation can be done among agents during games in order to decide formations, roles, tactics, etc. Then, our argumentation model is directed by the Contract Net Protocol [6], with which a specific agent called a coordinator (a kind of a playmaker) controls argumentation.

Our argumentation is divided into two types: reactive argumentation and deliberate argumentation. In the reactive argumentation, agents use only the information which is obtained in games. Issues to be argued in the reactive argumentation includes, who goes for a ball and dribbles, etc. In contrast, in the deliberate argumentation, agents make arguments from their own soccer knowledge and the knowledge that has been formed from the information obtained during games. Issues to be argued in the deliberate argumentation includes positioning or formation information of the other teams.

### 3.1 Deliberate Argumentation

The long argumentation is to be done at regular time intervals since too long and frequent arguments prevent the continuation of games. The argumentation protocol of the deliberate argumentation is as follows.

#### **Argumentation protocol for the reactive argument**

**Step 1.** The coordinator agent starts the long argumentation, and broadcasts an issue to all the other agents.

**Step 2.** Each agent makes its own argument about the issue, and sends it back to the coordinator agent.

**Step 3.** The coordinator agent receives the arguments, and sends one of them to all the other agents, asking to make counterarguments.

**Step 4.** The agents concerned make counterarguments, and send them back to the coordinator.

**Step 5.** The coordinator agent receives the counterarguments, and it compare the initial argument with those counterarguments. If some counterarguments defeat the initial argument, the coordinator agent send the counterarguments to all the other agents. And they repeat the same procedure as **Step 4**. Otherwise, go to **Step 6**.

**Step 6.** The coordinator agent looks for an argument which might be to be rebut next. If found, the coordinator agent sends it to all the other agents and go to **Step 4**. Otherwise, the long argumentation ends there. In the end, the coordinator agent sends the result of argumentation to all the other agents. All agents take an action following this result.

### 3.2 Reactive Argumentation

The long argumentation is not suitable for issues that require instantaneous decisions during games. In the reactive argumentation, agents argue about the issues on the motion of each agent by means of the information which is obtained during the soccer game. For example, they argue on which agent goes for the ball to dribble it. For the cases that agents have to reflect results of argumentation quickly and effectively in their next behaviors, we introduce the argumentation protocol of the reactive argumentation as follows.

#### Argumentation protocol for the reactive argument

**Step 1.** Prior to a game, we determine the preference relations for various information assumed to be obtained during the game, such as "the shorter the distance to a ball from a player is, the higher the chance to get it is".

**Step 2.** Playing agents ask the coordinator agent to start an argument on an issue relevant to the reactive argumentation. The coordinator agent seeks information necessary to this issue from other agents

**Step 3.** Each agent send required information to the coordinator agent.

**Step 4.** The coordinator agent sends to each agent a most relevant information of gathered information, which satisfies the preference criterion in **Step 1**.

## 4 Agent Architecture

Figure 1 depicts our agent architecture of arguing soccer agents.

The Communication class allows agents to communicate the environment information and commands with Soccer Server. The Data & Knowledge base class converts available information of the environment such as the position of a ball and visibility of agents into the form that the Main class can accept. The Main class decides the behavior of agents with information which is made by the Database class. It also sends the commands which are to decide the behavior of agents, to Soccer Server. The Action class provides the Main class with the

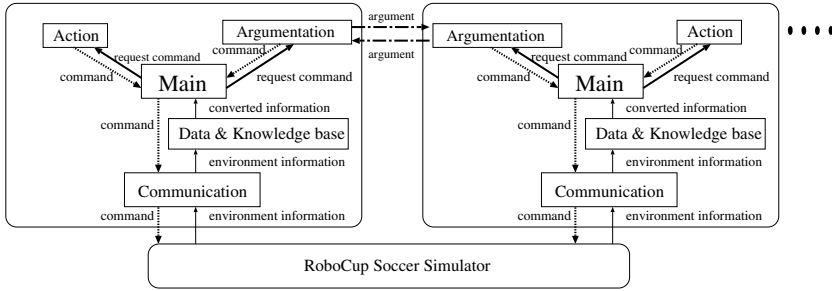


Fig. 1. Architecture of deliberate soccer agents powered with argumentation

commands which decide the behavior of agents. The Argumentation class allows agents to make arguments and counter-arguments during argumentation.

The Communication, Data & Knowledge base, Main and Action class are implemented by Java. The Argumentation class is implemented by Java and SICStus Prolog[7].

## 5 Some Technical Methods for Accelerating Argumentation

Accelerated argumentation is most crucial in building soccer agents with both reactivity and deliberation. If it takes a long time for them to argue and decide, it will turn out to reduce the effectiveness of the results of argumentation. In this paper, we consider two approaches to speeding up argumentation in the next two subsections.

### 5.1 Acceleration of the Argumentation Protocol

Dialogue trees [3][6] to decide the outcome of an argument are realized with the help of the argumentation protocol. Therefore, accelerating it most takes effect in speeding up argumentation. For this, we introduce two methods in which use of the argumentation protocol is restricted in one way or another.

**Avoidance of the same arguments.** We construct a dialectical tree so that agents do not repeat the same arguments or counter arguments as before in the same situation. This is a natural idea that can be seen in our daily argumentation or dialogue. Obviously, this helps to decrease the amount of time for making arguments or counter arguments.

**Introduction of resource-boundedness.** We humans live in resource-bounded world. Agents are no exception even if they live in a computer world. Unending arguments and nothing decided arguments must be avoided. We set a time limit for argumentation. If the time limit comes in argumentation, arguing agents stop arguing, obtaining a resource-bounded conclusion of argumentation.

## 5.2 Efficient Implementation

This involves in implementation techniques of arguing soccer agents. We introduce two techniques that deserve mentioning here.

**Thread programming.** Our soccer agents have two capabilities: argumentation with other agents, and reactive actions and decisions in the soccer environment. The former in general takes time compared with the latter. If two phases of argumentation and reactivity are executed sequentially, reactive actions and decisions are to be lost during a game. Therefore, two phases should be executed in parallel. In the sequential process, the soccer action phase is executed after the argumentation phase was finished. Because the argumentation phase takes a long time, the soccer phase is rarely executed. In the parallel process, the argumentation phase can be divided granularly, and the argumentation and soccer processes are to be executed alternatively. Therefore, the two phases should be expected to run in parallel. We achieved this using so-called thread technology. By this, arguing soccer agents were speeded up and run correctly.

**Multi-language programming.** Arguing soccer agents have argument engine implemented by SICStus Prolog. On the other hand, the agent communication is implemented in Java. The question is how two languages should communicate each other in the soccer agents system. If SICStus Prolog is called via the system from the agent program, it will take much time for the communication connection of two languages. Therefore, the agent program calls Prolog directly through SICStus Prolog Java library. This method helps agents to make arguments quickly.

## 6 Agent Communication

In RoboCup Soccer Simulator (RCSS) [1], official soccer agents communicate with each other by using 'Say' command. However, it is unfortunate that the Say command has a critical problem in introducing argumentation to arguing soccer agents. In RCSS, agents can hear only two messages by the Say command in 1 simulation step. In argumentation, therefore, agents might lose some counter-arguments if multiple counter-arguments are produced against arguments, and the Say command is used for sending them. Of course, synchronized agents could avoid this problem, in such a way that for example, when agent A says, others be quiet, and next when agent B says, others be quiet. In this way, if only one agent is allowed to say at any time in argumentation, argumentation could proceed well. But, obviously it will take a long time for agents to complete arguing in this method. Also, while the opposition communicates by the Say command, our agents can not communicate by the Say command. To be worse, agents may lose messages during argumentation since they communicate with RCSS by means of UDP.

To resolve these problems, we have decided to allow our agents to communicate directly with each other by TCP/IP, not the Say command. Put it



differently, we employed a method that somewhat deviates from the official rule of RoboCup, but one to invest a free communication capability to agents.

The messages communicated in our soccer agents have a form of  $\langle sender, receiver, subject, message\ type, content \rangle$ , where the message type denotes kinds of messages such as StartArg (for announcing an argument commencement), ReqArg (for asking agents to make arguments on an issue), ReqCounter (for asking agents to make counter-arguments), etc. , the content includes argumnets or counter-arguments.

## 7 Converting Environmental Information from Quantitative to Qualitative

Soccer agents receive the quantitative information about the environment from RCSS, such as the disatnce of a ball, the direction of other agents and so on. There is another kind of information in soccer games, such as the number of presence of the opposition in left side, the number of presence of a ball in the opposition side and so on. In either case, however, our soccer agents can't use directly the quantitative information in argumentation since they use qualitative information in a form of rules. That is, they can't use directly soccer information in argumentation.

Our arguing soccer agents need to convert the quantitative soccer information into the quantative information or rules. Let see an example of quantitative information in a soccer game. Suppose that we have the accumulated soccer information concerning the number of presence of the opposition in a certain simulation time, such that the ratios of their positions are 100 in the left side, 80 in the right side, and 150 in the center. Then, we infer that the presence of the opposition tends to be in the center, based on these figures, and then our soccer agents produce the rule.

$$enemy\_presence(center) \Leftarrow .$$

meaning that the opposition tends to take the positioning in the center area. Actually, we employ 500 simulation times for collecting soccer information that is needed for changing game strategies such as formation change.

## 8 Two Argument Examples

### 8.1 An Example of the Reactive Argumentation

Figure 2 and 3 display the screen shots before and after a reactive argumentation. In Figure 2, all agents are about to go for the ball in the center circle. But in Figure 3, the only one agent is going for it. This is because agents have finished arguing who should go for the ball with the reactive argumentation. Thus, it helped agents avoid packing into a ball, and enabled agents to do the coordinated action which resulted in a group decision in a soccer game. It should be noted that it is not a centralized but a distributed and autonomous coordination that soccer-playing agents take.

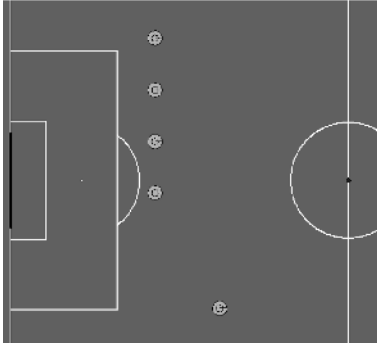


Fig. 2. Before argumentation

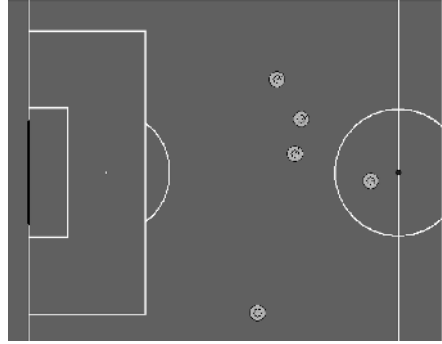


Fig. 3. After argumentation

### 8.2 An Example of the Deliberate Argumentation

Figure 4 and 5 display the screen shots before and after a deliberate argumentation. The left side team is our arguing soccer agents. The right side team is soccer agents without argument ability. In Figure 4, the formation of our agents is 3-5-2. After a goal in Figure 4, they argue about the formation, so as to win the next game. They use the accumulated information (Section 7) and argue under a time limit (Section5). For example, let us consider again the example taken in Section 2. Suppose one of agents made the following argument *Arg*.

$$\begin{aligned}
 Arg = \{ & \neg enemy\_offensive \Leftarrow . \\
 & enemy\_front(more) \Leftarrow \\
 & formation(4 - 4 - 2) \Leftarrow \neg enemy\_offensive \wedge enemy\_front(more). \}
 \end{aligned}$$

The conclusion of *Arg* is that "Our formation must be 4-4-2.". After the argumentation, if it is accepted as a justified argument among agents, they accept its result, and change their formation from the previous 3-5-2 formation for example into 4-4-2 one for the next game as in Figure 5. The experimental result of such a formation change was 3-0 between a soccer agent team who changed

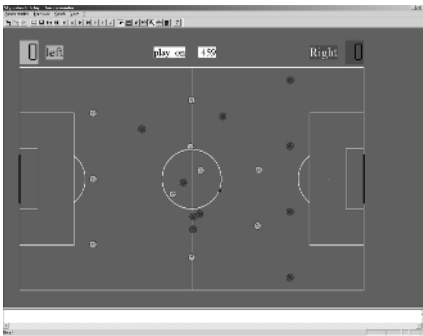


Fig. 4. From 3-5-2 formation

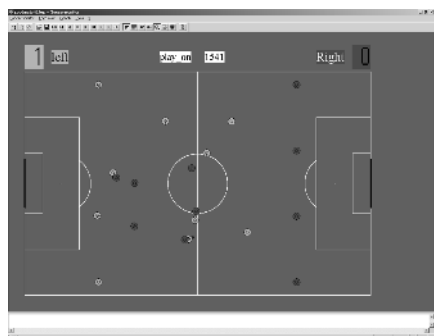


Fig. 5. To 4-4-2 formation

a formation and another team who did not. This shows a fruitful effect of the introduction of argumentation into soccer agents.

## 9 Experimental Evaluation

First we tried the 10 games of soccer agents without argument ability v.s soccer agents without argument ability. Then, the results are 2 wins, 6 evens, and 2 losses. Next we tried the 10 games of arguing soccer agents v.s soccer agents without argumentation. The results are that arguing soccer agents get 4 wins, 4 evens and 2 losses. Consequently, it would be fair to say that argumentation surely strengthens the soccer agents. It is noted that the fluctuation of the win and loss comes from that given to the RoboCup Soccer Simulator [1].

**Table 1.** Game results

VS.	Agents without argumentation
Agents with argumentation	4 wins, 4 evens, 2 losses
Agents without argumentation	2 wins, 6 evens, 2 losses

## 10 Conclusion

In this paper, we attempted to introduce argumentation to soccer agents. They might say that this is a first, but reckless attempt. However, we could give many reasonable answers to problems proper to the arguing soccer agents. Furthermore, we could not successfully implement it without techniques accelerating argumentation in one way or another. The experimental results also showed a potential and practical possibility of resource-bounded argumentation even in a typical reactive agent system such as soccer agents. The video clip and storyboard of deliberate soccer agents powered with argumentation are accessible at our web site [4].

## References

1. M. Chen et al. *RoboCup Soccer Server Users Manual*. <http://www.robocup.org/>, 2002.
2. C. I. Chesnevar, G. Maguitman, and R. P. Loui. Logical models of argument. *ACM Computing Surveys*, 32:337–383, 2000.
3. T. Nisikata. Arguing soccer agents. *Master’s Thesis, Niigata University*, 2005.
4. T. Nisikata. *Video clips and storyboard of deliberate soccer agents powered with argumentation*. <http://www.cs.ie.niigata-u.ac.jp/~nisikata/research.html>, 2005.
5. H. Prakken and G. Sartor. Argument-based extended logic programming with defeasible priorities. *J. of Applied Non-Classical Logics*, 7(1):25–75, 1997.
6. H. Sawamura and S. Maeda. An argumentation-based model of multi-agent systems. In H. Jaakkola, H. Kangassalom, and E. Kawaguchi, editors, *Information Modeling and Knowledge Base XII*, pages 137–150. IOS Press, 2001.
7. SICStus. Sicstus prolog manual. <http://www.sics.se/isl/sicstuswww/site/documentation.html>, 2004.

# Agent-Oriented Probabilistic Logic Programming with Fuzzy Constraints\*

Jie Wang and Chunnian Liu

Key Laboratory of Multimedia and Intelligent Software,  
Beijing University of Technology, 100022, Beijing P.R. China  
{wj, ai}@bjut.edu.cn

**Abstract.** Agent-based computing is an important research area. Agent and multi-agent system can be used to model real systems in complex and dynamic environments. However, most of them assume that there is no uncertain and fuzzy information in an agent's mental state and environment. In this paper, we remove this unrealistic assumption and propose a new agent programming language which allows agent programs to effectively perform with fuzzy knowledge under uncertain environment, and to dynamically adapt with changes of the environment. This language presents a new and practical approach to solve fuzzy constraints in uncertain environment which consists of three components for programming agent: uncertain belief updating, goal selection and revision and uncertain practical reasoning.

## 1 Introduction

Agents are software or hardware systems that can autonomously take actions based on changes in their environment. In modeling rational agents, modeling agent's attitudes(mental states) as Belief, Desire, Intention (BDI agent) has been the best known approach. There has been number of formalizations and implementations of BDI agents such as [14,4,6,12]. Belief, desire, intention attitudes of a rational agent represents the information that the agent has about the environment, the motivation of what it wants to do, and finally the plans that the agent intends to execute to achieve its desired goals. This sort of agents have a set of goals and try to realize their goals by means-end reasoning or practical reasoning according to its belief and plans.

Although, researchers have tackled most issues of BDI agents from logical systems to logic programming [5,14]. But most of them assume there is no uncertainty in an agent's mental states and its environment. The issue of acting with uncertain knowledge about the environment has little been addressed in BDI agent programming languages. The problem normally faced in open-dynamic environment is to adapt with changes of environment both to react to changes and to adjust strategies to achieve previously adopted goals. My previous paper[16] remove the unrealistic assumption and proposed an agent-oriented probabilistic logic programming

---

\* This work is supported by the NSFC major research program under Grand No. 60496322 and Open Foundation of Key Laboratory of Multimedia and Intelligent Software (Beijing University of Technology).

language which can deal with uncertain information. But uncertainty is not the only imperfect information, sometimes we have to face fuzziness. For example, to evaluate cars, the following attributes must be put into consideration: price, gas mileage, safety, legroom, workmanship and style. For the example in [16], when an agent try to destroy an object, how to chose the action not depending on the action's maximum expected utility [16], but also the environment and the attributes of an object are also very important factors(e.g, if weather is heavy fog or the object seems easy to destroy).

In order to deal with this kind of situation, i.e., fuzziness in uncertain environment, in this paper, we further endow our probabilistic logic programming with fuzzy constraints, develop fuzzy-constraint based multiple attributes decision for probabilistic logic programming and propose a new agent programming language which allows agent programs to effectively perform with fuzzy knowledge about the uncertain environment, and to dynamically adapt with changes of the environment. This language presents a new and practical approach to solve fuzzy constraints in uncertain environment which consists of three components for programming uncertain, fuzzy BDI agent: the first one is the function to update uncertain belief based on probability theory; the second one is the goal selection and revision based on multiple attributes decision theory; and the third one is the propagation of uncertainty and fuzziness along practical reasoning.

## 2 Primaries

The uncertain beliefs of an agent can be expressed in first-order formulae from a language  $L$  consisting of a finite set of predicate symbols and finite set of variable symbols. Let  $\rho$  be a sub-interval of  $[0,1]$ , the basic elements of the language are given by a signature  $\Sigma = (P, F, C, A, \rho)$ , where  $P$  is a set of predicate symbols,  $F$  is a set of function symbols,  $C$  is a set of constants,  $A$  is a set of action symbols,  $\rho$  is a set of probabilistic constants. Terms and atoms of a language  $L$  are defined as usual from a signature  $\Sigma$  and an infinite set of variables  $TVar$ . The notion of variable substitution, unifier, most general unifier, and variant are defined as usual [8]. The set of atoms is denoted by  $At$ .

A  $\rho$ -atom is an expression of the kind  $E/\rho$  with an atom  $E$  and interval probability  $\rho$  which is called the probability that  $E$  occurs. When  $\rho = [1, 1]$  (abbreviated for  $E/1$ ), it means  $E$  is true; when  $\rho = [0, 0]$  (abbreviated for  $E/0$ ), it means  $E$  is false; and when  $\rho \subset (0, 1)$ , it means  $E$  probably occurs. Intuitively,  $E/\rho$  means that the probability of  $E$  being true lies in the interval  $\rho$ . The set of  $\rho$ -atoms is denoted by  $\rho\text{-}At$ .

Two basic kinds of goals that an agent tries to achieve: goals to do some action and goals to achieve some state of affairs. More complicated ones can be constituted from the two basic kinds in the following manner:

**Definition 1.** Let  $\Sigma = (P, F, C, A, \mu)$  be a signature, and  $Gvar$  an infinite set of goal variables ranging over goals. The set of goals  $L^g$  is inductively defined by:

1.  $A \subseteq L^g$ , called basic actions.
2.  $\rho\text{-}At \subseteq L^g$ . In particular,  $At \subseteq L^g$ .

- 3. If  $\varphi \in L$ , then  $\varphi? \in L^g$ .
- 4.  $Gvar \subseteq L^g$ .
- 5. If  $\pi_1, \pi_2 \in L^g$ , then  $\pi_1; \pi_2, \pi_1 + \pi_2$ .

Basic actions  $a \in A$ , achievement goals  $\varphi \in p\text{-}At$ , and test goals  $\varphi?$  are the basic goals. Basic actions are updating operators on the uncertain belief base of an agent. Achievement goals  $\varphi$  are goals to achieve a state where  $\varphi$  holds. Test goals  $\varphi?$  are checks performed on the belief base to see if a proposition  $\varphi$  holds. More complex compositions can be built from basic goals, by using the program to construct sequential composition( $;$ ), and nondeterministic choice( $+$ ). composition( $||$ ). About the action/goal selection, we will discuss in next section.

### 3 Action Selection with Fuzzy Constraints

Consider a scenario of scout Rob which search objects in a two-dimensional grid world. It intends to destroy military-objects that it can, if it cannot, it will inform main force to do. Sometimes, it cannot make sure an object is a military or a civil one. Suppose it find three objects  $a, b$  and  $c$  simultaneity, the distance that Rob to  $a, b, c$  are 20m, 40m and 60m, respectively. The probabilities that each object is a military object are as follows:  $p(a \text{ is a military object})=0.4$ ,  $p(b \text{ is a military object})=0.75$ ,  $p(c \text{ is a military object})=0.85$ .

For each object, consider three attributes: the damageable, definition and volume as shown in Table 1. Now the action selection of Rob not depend on the

**Table 1.** Fuzzy Attributes of the Objects

Object	Damageable	Definition	Volume
a	difficult	very clear(sunshine)	large
b	normal	clear	small
c	easy	fuzzy (heavy fog)	middle
weight	0.5	0.3	0.2

maximum expected utility [16], but also multiple attribute of objects have to be put into consideration! we integrate a fuzzy-constraint-based utility-theory approach for modeling . That is, agent does either  $e_i$  or  $e_j$  ( $i = 1, 2, \dots, n$ ) based on not only the uncertain beliefs, but also the multiple attributes of each object.

We briefly review the basic idea of multiple attribute decision making [7] refers to making selections among courses of action in the presence of multiple, usually conflicting attributes. It can be concisely expressed in matrix format as:

$$D = \begin{matrix} A_1 \\ A_2 \\ \vdots \\ A_m \end{matrix} \begin{pmatrix} x_1 & x_2 & \cdots & x_n \\ x_{11} & x_{12} & \cdots & x_{1n} \\ x_{21} & x_{22} & \cdots & x_{2n} \\ \vdots & \vdots & \dots & \vdots \\ x_{m1} & x_{m2} & \cdots & x_{mn} \end{pmatrix} \tag{1}$$

$\omega = (\omega_1, \omega_2, \dots, \omega_n)$

where  $A_i, i = 1, 2, \dots, m$  are possible course of actions (referred to as alternatives);  $x_j, j = 1, 2, \dots, n$  are attributes with which alternative performances are measured;  $x_{ij}$  is the performance rating of alternative  $A_i$  with respect to attribute  $x_j$ ,  $\omega = (\omega_1, \omega_2, \dots, \omega_n)$  are the relative importance of attributes (decision weights), alternative performance rating  $x_{ij}$  can be crisp, fuzzy, and/or linguistic. In this paper, we restrict the ratings as crisp numbers in  $[0, 1]$ . The relative importance is usually given by a set of weights which are normalized to sum to one. That is, in the case of  $n$  attributes, weight set is  $\omega^T(\omega_1, \omega_2, \dots, \omega_n)$  and  $\sum_{i=1}^n \omega_i = 1$ . The weights can be assigned by the decision maker directly.

Basically, the chosen alternative should have the shortest distance from the ideal solution and farthest distance from the negative ideal solution. We resolve the problem through the method of TOPSIS [1,7]. I.e., for rating attributes, first quantity them by Bipolar Scaling method [7], and then transfer the qualitative attribute ratings to quantitative ones; Then calculate the relative closeness (utility) to the ideal solution; Finally, rank the preference order.

The relative utility represents how well one alternative satisfies the decision maker’s requirement. The alternatives with higher utilities are preferred by the decision maker. Advantages of the method is simple, easy to use and understand.

So if there is a crisp constraint  $R_i, i = 1, 2, \dots, n$  with priority  $\rho(R_i) = \delta_1$ , and fuzzy constraint  $R_m$ , i.e., fuzzy multiple attribute, has priority  $\rho(R_m) = \delta_2$ . We can calculate the overall acceptability to each object to decide if acting according to the overall satisfaction formula in [11]. The overall acceptability threshold  $\tau$  means: if the overall acceptability of an object is not less than the threshold  $\tau$ , the object is acceptable as a solution; otherwise, it is not.

Now we examine the example above. Firstly, by the method of TOPSIS, calculate the relative closeness (utility) to the ideal solution  $U(a) = 0.2251, U(b) = 0.2251, U(c) = 0.2251$ ; Then rank the preference order and obtain:  $c > b > a$ . This means that  $c$  is the most satisfactory object to the attributes. Suppose when agent *Rot* destroys an object, it must satisfy two crisp constraints:

- $R_1$  : “the probability that  $x$  is a military object  $\geq 0.7$ ” with priority  $\rho(R_1) = 0.65$
- $R_2$  : “distance  $\leq 50$ ” with priority  $\rho(R_2) = 0.50$

Object  $a$  is not satisfied with crisp constraint  $R_1$ ,  $c$  is not satisfied with crisp constraint  $R_2$ . Given the acceptability threshold  $\tau$  is 0.7. According to the overall satisfaction formula in [11], the overall acceptability that agent to object  $b$  is:

$$\alpha_\rho(v_b) = \min\left\{\frac{\rho(R^f)}{\rho_{\max}}\right\} \diamond \mu_{R_f}(v_{var}(R_f) \mid R_f \in C_f) = 0.7310 > 0.7$$

Noticing the overall acceptability is greater than 0.7 (threshold), so agent *Rob* can destroy the object  $b$  and do nothing to the object  $a$  and  $c$ . In addition, the programming language for BDI-agent includes variables which range over goals. These variables can be used for reflective reasoning and for communication.

## 4 Uncertain Practical Reasoning Rules

To achieve its goals agents have to find the means for achieving them, sometimes may have to revise its goals. This kind of reasoning is called practical

reasoning. To perform this type of reasoning agents use a set of practical reasoning rules:

**Definition 2.** *Let  $\varphi \in L, \pi, \pi' \in L^g$ . Then an uncertain practical reasoning rule is in the form of*

$$\pi \leftarrow \varphi | \pi' \in L^p, \quad \text{where}$$

- $\pi, \pi', \varphi$  is called the head, the body and the guard of the rule respectively;
- the free variables in the head are called the global variables of the rule; and
- all variables in a rule that are not global are called local variables.

Since  $\varphi, \pi, \pi'$  are  $\rho$ -atoms which represent uncertainty (the overall acceptability is also represented in the body), practical reasoning rules can actually represent and reason under uncertainty and fuzzy information. The guard can be used to specify the context in which the rule might be applied and to retrieve data from the set of beliefs.

The function of a practical reasoning rule is twofold. First, a rule can specify the means to achieve a particular goal under uncertainty. Uncertainty is involved in both of the head and the body of the rule. A plan for getting agent at a specific position, see [16].

**Example 1.** *Agent Rob wants to destroy military objects (Mobj) in its world. A plan rule which specifies how to achieve this goal is the following.*

$$\text{destroy-Mobj} \leftarrow P(\text{Mobj}) \geq 0.7 \wedge (\text{distance} \leq 50) | \alpha_\rho(v_x) \geq \tau, \text{destroy-Mobj}.$$

*This rule specifies a plan to achieve destroy-military-object. The plan is to destroy an object that  $\alpha_\rho(v_x) \geq \tau$ , destroy that military object, and then recursively start destroying military objects again. The guard retrieves crisp constraints.*

The second purpose of practical reasoning rules is the revision of goals. Basically, there are two types of situations in which a rational agent wish to revise one of his goals, i.e., in case of failure, or in case a more optimal strategy can be followed. For the former, the details are discussed in [16]; For the latter, we use the formalization of decision-theoretic default[16] and show an instance. For this case, we modify the definition of decision-theoretic default to deal with fuzzy multiple attributes decision, i.e., if  $e$  is a formula in first-order language  $L$ ,  $A = \{a_1, \dots, a_n\}$  is a set of possible alternative actions, and  $a \in A$ , then

$$e \rightarrow_A a \quad \text{if } \alpha_\rho(v_x) \geq \tau,$$

where  $\tau$  is agent's overall acceptability threshold. So, goal update depend on not the crisp constraints and fuzzy constraint, but the overall acceptability threshold.

**Example 2.** *For the example in Section 3, suppose  $\tau$  is changed to 0.8. The overall acceptability  $\alpha_\rho(v_a), \alpha_\rho(v_b)$  is also 0, agent do nothing to object a and c. But  $\alpha_\rho(v_b) = 0.7310 \leq 0.8$ , Rob will inform main force to destroy object b. Then agent go ahead. Suppose now the crisp constraint  $R_2$  are all satisfied by*



these objects: (uncertain belief is not changed). So  $\alpha_\rho(v_c) = 0.9285 > 0.8$ , a more optimal strategy for Rob is to revise its current goal for destroying object  $c$ . The following rule makes this type of goal-revision possible:

$$G : \text{Rob}(x, y) \leftarrow (\alpha_\rho(v_c) > \tau) \wedge (\text{distance} \leq 50) \mid \text{destroy}(c)$$

The rule applies when agent has a goal of doing some (probably empty) list of actions  $G$ , after that the distance that agent to the object is changed, the overall acceptability will also changed. In the case the rule makes it possible to revise the old goal by replacing it by a new goal of destroying the object  $c$ .

Notice that there are two special cases of uncertain practical reasoning rule: the rule with an *empty body* and the rule with an *empty head* [16].

## 5 Probabilistic Agent Programs

Assume beliefs are updated by uncertain information, and goals are updated by execution and revision. The uncertain practical reasoning component is encoded in uncertain practical reasoning rules. Thus, an agent's beliefs and goals can change but practical reasoning rules and basic actions are fixed. Beliefs and goals constitute the mental state of an agent. Formally, we define:

**Definition 3.** A mental state is a pair  $\langle \Pi, \sigma \rangle$ , where

- $\Pi \subseteq L^g$  is a set of goals called a goal base, and
- $\sigma \subseteq L$  is a set of beliefs called a belief base.

The dynamics of behavior of an agent, therefore, is fully specified if the semantics of basic actions is given and the mechanisms for executing goals and applying rules are defined.

**Definition 4.** The semantics of basic actions  $A$  is given by a transition function  $\mathcal{T}$  of type:  $B \times B \rightarrow \varphi(A)$ . Let  $a \in A$ . We use the following notational convention, and write:  $\langle \sigma, \sigma' \rangle a$  for  $a \in \mathcal{T} \langle \sigma, \sigma' \rangle$ .

Agent is capable of performing five basic actions. About the operational semantics of action ‘west’, ‘east’, ‘north’, ‘south’ and ‘destroy’ is given in [16]. Thus, to program an agent is to specify its initial mental state, the semantics of the basic actions that the agent can perform, and a set of uncertain practical reasoning rules. This is formally represented in the next definition.

**Definition 5.** A probabilistic agent program is a quadruple  $(\mathcal{T}, \Pi_0, \sigma_0, \Gamma)$ , where

- $\mathcal{T}$  is a basic action transition function, specifying the effect of basic actions.
- $\Pi_0, \sigma_0$  are the initial goal base and the initial belief base respectively.
- $\Gamma$  is a PR-base.

**Example 3.** *The agent program for agent Rob is the following:*

- $\mathcal{T}$  is defined for the basic actions in Example 3,
- The initial goal base is given by:  $\{\text{destory-military-object}\}$ .
- The initial belief base is given by  
 $\{\text{Rob}(0, 0), \text{object}_a(0, 20)/0.4, \text{object}_b(0, 40)/0.75, \text{object}_c(60, 0)/0.85\}$
- The PR-base contains the PR-rules as defined in examples 1 and 2.

*The agent program works according to the following steps. Firstly, it will test if the overall acceptability that the agent to the object is more than its acceptability threshold  $\tau$ . If yes, it will act according to the rule in Example 1; otherwise agent will looking for other object. Secondly, if agent's belief change in the processing of go ahead, it will revise the goal according to the rule in Example 2.*

## 6 Related Work

There have been several well known work [14,4,5,2] for agent programming languages from Shoham's very first work [15] in 1993. For example, Rao [14] aim to implement of BDI agents in a logic programming style. It is an attempt to bridge the gap between logical theories of BDI and implementations without uncertainty. The work of Hindriks [4] is a rule-based language which is similar to our work, a configuration of an agent consists of a belief base, a goal base, and a set of practical reasoning (PR) rules, so an agent can perform in highly dynamic environment, but uncertainty is not included. The work of Levesque [5] is an extension of situation calculus, it is a concurrent language for high-level agent programming and uses formal model semantics, An assumption is that the environment changes only if a agent performs an action. This strong assumption is not required in our work. Furthermore, the extensive advantage of our work in comparing with [14,4,5] is our work allows agents to act with fuzzy constraint in uncertain environment. although Dix and Subrahmanian [2] gave a proposal for programming probabilistic agents, unlike our work, fuzzy constraint is not discussed and theirs is not based on BDI concepts in their work.

On the other side, Several researchers extend the BDI model to the situation of uncertainty [12,6]. However, all of them cannot handle the issues of goal and action selection and revision under uncertainty and the issue of uncertain practical reasoning. Nevertheless, these limitations are removed in our work. In addition, although fuzzy information is put into consideration in the agent systems [9,3], no concepts of fuzzy constraint, probabilistic uncertainty and BDI agent are involved in an agent systems.

## 7 Conclusion and Further Work

Uncertainty and fuzziness is unavoidable in agent systems. This paper proposes a new agent programming language which allows agent programs to effectively perform with fuzzy knowledge under uncertain environment, and to dynamically adapt with changes of the environment. This language consists of three

components for programming agent: uncertain belief updating, goal updating and uncertain practical reasoning.

Further work would be extended from this work. First, we will present a formal semantics which can deal with uncertain and fuzzy information for BDI agent, then give an arithmetic for the example illustrated in this paper. Second, there would be an extension of this framework with multi-agent environment, where agent's intentions are influenced by others' beliefs and intentions.

## References

1. Shun-Jen Chun, Ching-Lai, Hwang and Frank P.Hwang. Fuzzy multiple attribute decision Making: Methods and Applications. Springer-Verlag, 1992.
2. Dix J., Subrahmanian V S. Probabilistic agent programs. *ACM Transactions on Computational Logic*, 1(2), pp.207-245, (2000).
3. He M., Jennings N.R., and Prgel-Bennett A. A heuristic bidding strategy for buying multiple goods in multiple English auctions. *ACM Transactions on Internet Technology*, (2006). (To Appear)
4. Hindriks K.V., de Boer F.S. , van der Hoek W., and Meyer J.J.C., Formal Semantics of an Abstract Agent Programming Language, *Proceedings of International Workshop on Agent Theories, Architectures, and Languages (ATAL '97)*, LNCS 1365, pp. 215-229, Springer, (1998).
5. Levesque H., Lesperance R.R., Y.F.L.and R.S. Golog: A logic programming language for dynamic domains. *Journal of Logic Programming*, 31:59.84, (1997).
6. Li Y. and Zhang C. Information fusion and decision making for utility-based agents, *Proceedings of the Third World Multi-conference on Systemics, Cybernetics and Informatics and the Fifth International Conference on Information Systems Analysis and Synthesis*, (1999).
7. Li R. Fuzzy multiple criteria decision: theory and application. Science Press, (2002).
8. Lloyd J. *Foundations of Logic Programming*, Springer, (1984).
9. Luo X., Zhang C. and Leung H.-f. Information sharing between heterogeneous uncertain reasoning models in a multi-agent environment: A case study, *International Journal of Approximate Reasoning*, 27(1), pp. 27-59, (2001).
10. Luo X., Zhang C., and Jennings N.R. A hybrid model for sharing information between fuzzy, uncertain and default reasoning models in multi-agent systems, *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 10(4), pp. 401-450, (2002).
11. Luo X., Jennings N.R., Shadbolt N., Leung H.F. and Lee J.H.M. A fuzzy constraint based model for bilateral, multi-issue negotiation in semi-competitive environments, *Artificial Intelligence*, 148, pp.53-102, (2003).
12. Parsons S. and Giorigini P. An approach to using degrees of belief in BDI agents, *Information, Uncertainty and Fusion* , pp. 81-92, Kluwer, (2000).
13. Poole D. Decision theoretic defaults, *Proceedings of the 9th Biennial Canadian artificial Intelligence Conference*, pp.190-197, (1992).
14. Rao A.S. AgentSpesk(L): BDI agents speak out in a logic computable language, *Agents Breaking Away*, pp. 42-55, (1996).
15. Shoham Y. What we talk about when talk about software agents, *IEEE Intelligent Systems*, pp. 28-31, March/April, (1999).
16. Wang J., Ju S. and Liu. C. Agent-Oriented Probabilistic Logic Programming. *Journal of Computer Science and Technology*, vol(21) 3, pp.412-417, (2006).

# An Agent-Based Adaptive Monitoring System

Sungju Kwon and Jaeyoung Choi

School of Computing, Soongsil University,  
1-1 Sangdo-5Dong, Dongjak-Gu,  
Seoul, 156-743, Korea  
{lithlife, choi}@ssu.ac.kr

**Abstract.** In Grid systems, which have become more and more bigger and more complex, it is critical to utilize heterogeneous resources efficiently. In this paper, we designed and implemented an agent-based monitoring system, which consists of dynamically controllable agents. The structure of the monitoring system is divided into three layers to archive independence among communication protocol, message interpretation, and actual tasks. Components of each layer can be replaced with other components whenever they are required. To reduce development times for monitoring application in distributed systems, four monitoring agents were provided, which help to reuse legacy applications through user query.

## 1 Introduction

In distributed systems such as Grid [1], enterprise computing, and ubiquitous computing, one of the big issues is management of services, processes, and system resources, which were geographically distributed and accessed heterogeneously. Service Oriented Architecture (SOA) [2, 3], which is designed by Web Services Architecture Working Group in W3C, was proposed to address these problems, and the architecture became the basic structural concept of recent distributed systems. Web Services Architecture Working Group defines SOA as a form of distributed systems architecture with properties such as logical view, message orientation, description orientation, granularity, network orientation, and platform neutral. Because of the characteristics of these properties, heterogeneous resources can cooperate with each others without constraints such as implementation languages, operating platforms, and communication protocols. However, the architecture still has a problem using the distributed resources. Because of the heterogeneous characteristics and frequent changes of the resource situation, many systems were distressed with efficient use of the distributed resources. If one can apply dynamically controllable agent-based systems [4, 5] in distributed environments, it will be possible to use the heterogeneous resources more efficiently and effectively. Each agent can be loaded, unloaded onto a particular resource, or can be moved to another resource depending on environmental changes.

In this paper, we present an agent-based monitoring system, which can control the lifecycle of agents dynamically. The monitoring system is separated into three independent layers based on the role: communication protocols, message interpreters, and

monitoring tasks. The independence among three layers can reduce the time for agent development, and can make easy to manage distributed systems.

## 2 Related Works

Grid is the extension of traditional distributed systems to share resources among virtual organizations. There are, however, some difficulties using resources in Grid. Because of the heterogeneity and the dynamic change of resources, the monitoring of resources is a crucial job in Grid.

Grid Monitoring Architecture (GMA) [6] has been proposed by Grid Monitoring Architecture Working Group (GMA-WG) [7] of GGF to encourage monitoring functionality for Grid systems. Many Grid monitoring systems have been implemented based on this architecture. Java Agents for Monitoring and Management (JAMM) [8] is a wide-area GMA-like system that was developed at Lawrence Berkeley National Laboratory. This system uses sensors to collect and publish computer host information. Clients can control the execution of remote sensors and receive monitoring data in the form of events. Relational Grid Monitoring Architecture (R-GMA) [9] was developed within the European DataGrid [10] project as a Grid information and monitoring system. R-GMA is an implementation of the GMA specifications and provides a relational model for information processing. It provides an information transport layer for host, network, service, and application monitoring data. These systems and other monitoring systems [11] are well suited for the specialized domain. These monitoring systems, however, still have a problem in that they are not flexible or adaptable for rapidly changed resources in Grid.

To solve these problems, agent-based systems can be used. There are many papers to describe agent-based monitoring systems. Because of the use of agents, these agent-based monitoring systems are easy to load, unload, move, or replace the agent on any nodes. The systems, however, do not have layered architecture, so the transparency of the network or message cannot be provided, which is necessary for distributed systems.

In this paper, we present a more efficient way to implement a monitoring system with multi-agents on the layered framework. This system can be used to build various monitoring applications with much less effort in distributed systems.

## 3 An Agent-Based Adaptive Monitoring System

The design and implementation of the agent-based adaptive monitoring system for distributed applications are described in detail in this section. The implementation of the agent-based monitoring system is based on a multi-layered framework, which comprises three different layers: the Communication Layer, Interpreter Layer, and Agent Management Layer. Fig. 1 shows the overall layout of the framework.

The **Communication Layer** acts as a manager for protocol components. These components receive remote users' messages and encapsulate the messages into a pre-defined *Message* object, which is defined internally in the monitoring system. The **Interpreter Layer** transfers the *Message* using an appropriate Query component.

A user sends a message with information of query type, so the Interpreter Layer can select the corresponding query component easily. The **Agent Management Layer** controls the lifecycle of agents that actually execute a task according to the user's request. The interpreted messages are delivered to this layer with their abstract service names. Four basic agents are implemented to provide monitoring services. These agents gather resource information from host nodes in four different ways and provide services to match user requests. Every component on each layer can be dynamically activated, deactivated, and moved by the system administrator's request.

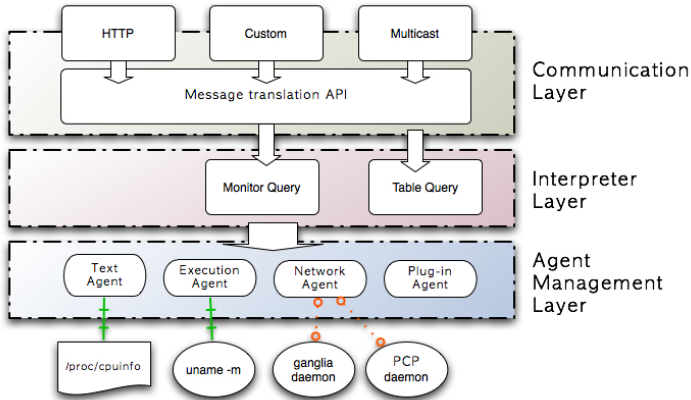


Fig. 1. The Architecture of our multi-layered framework

### 3.1 Communication Protocol

To provide easy replacement of the communication protocol without affecting other layers, protocol interfaces for components were provided. An application can choose an appropriate communication protocol component in a specific environment. Each component receives requests from users and interprets them through its protocol. Then the component makes a Message object and encapsulates the request messages in the Message object. A Message object contains properties such as user ID, password, token, query type, command, code, and actual message. For basic communication facility of the monitoring system, we developed three communication protocol components: HTTP, Multicast, and Custom protocol component.

### 3.2 Query Interface for Interpreting Request Message

The Interpreter Layer does not need to worry about physical communication, which is performed in the Communication Layer. The Interpreter Layer only needs to care about actual meaning of the message. Two query components exist: **Monitor Query** and **Table Query**. Monitor Query looks similar to a SQL's SELECT statement. A user can query monitoring information using a SELECT statement of Monitor Query. The format of this SELECT statement could be as follows:

```
SELECT field1 [, field2, ...] FROM service name [DOMAIN node name]
[WHERE condition [AND | OR condition]]
```

The name of the actual monitoring agent is described in the FROM clause. The Interpreted message will be delivered to a monitoring agent in the form of hash table. If the DOMAIN clause has a particular node name, the request message will be delivered only to an agent in the specified host. The Monitor Query can be used to request monitoring information from monitoring agents. The **Table Query** consists of a pair of data: variable and value. This query information is used to check the heartbeat signal of each node as well as the resulting information of the monitoring request.

### 3.3 Monitoring Agents

To reuse legacy applications easily, four basic monitoring agents were defined: text sensor agent, execution sensor agent, network sensor agent, and plug-in sensor agent. Through these four agents, the only thing that users have to do is to customize queries depending on the tasks. The **Text Sensor Agent** reads a text file from a local node and sends back the filtered result to the user. The **Execution Sensor Agent** executes an application on a local node and gathers the results. The **Network Sensor Agent** communicates with other applications using TCP/IP. With this agent, users can specify the protocol sequences to gather results. Currently, the network sensor agent can process simple protocols and will be improved in the near future. The **Plug-in Sensor Agent** manages agents developed by users. This agent provides a caching facility to reduce development time of the monitoring agent.

## 4 Implementation of an Agent-Based Monitoring System

Based on the framework described in previous section, an agent-based monitoring system was implemented. This system gathers recently updated information from each node through multicast protocol automatically. In this system, a system administrator does not need to update information about newly appended nodes. To provide this feature, three kinds of monitoring management agents were installed on each node. Fig. 2 shows the layout of the implemented monitoring system.

The **Server Broker** collects information of active nodes with the help of the Server Proxy. This agent sends a heartbeat signal to Server Proxy. The Server Broker has role of maintaining the information of every active Server Proxy and NodeInfo Agent. Another role of the Server Broker is to deliver users' requests to appropriate monitoring agents. The **Server Proxy** accepts the heartbeat signal from the Server Broker and delivers the message through multicast protocol to all active NodeInfo Agents. The **NodeInfo Agent** sends the local nodes' information to the Server Broker in response to the multicast message of the Server Proxy. The node information consists of the local network address, the type of protocols, the type of query interfaces, and a list of sensor agents.

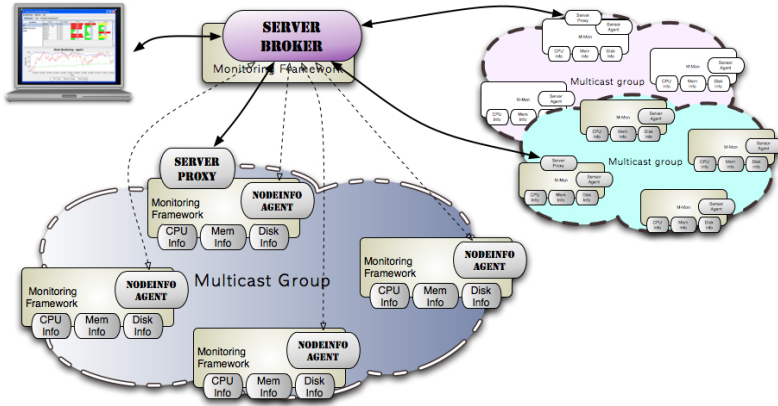


Fig. 2. Layout of the agent-based monitoring system

We developed a client application that displays basic information of nodes such as CPU load, memory usage, and disk usage. In Fig. 3, a user can select monitoring groups using a list box at the upper left corner of the user interface. The advantage of this system is that a user only needs to specify the addresses of the Server Broker and each Server Proxy. After startup, the Server Broker agent automatically updates lists of each active Server Proxy and NodeInfo Agent.

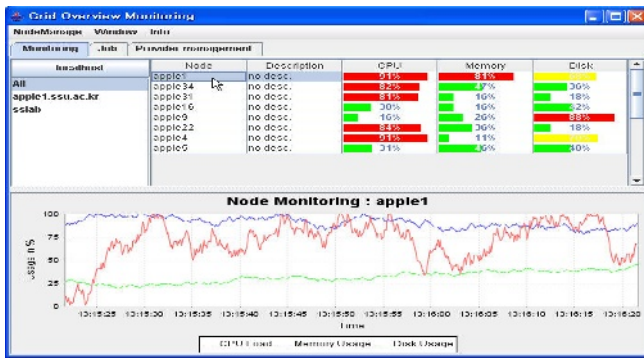


Fig. 3. An example of client application for user interface

## 5 Conclusion

In this paper, an agent-based monitoring system has been presented. This system consists of three layers for the developer can easily replace a part of the application without interference from other various applications. Because layers are separated from each other, an application deports. The agent-based monitoring system has the following features. First, the monitoring system uses a component-based architecture to provide reconfigurability. Second, the system separates functions into three main



layers: communication, interpretation, and agent management layers. Third, the monitoring system provides four basic monitoring agents that reuse legacy resources. Finally, the system provides a basic architecture to build an actual monitoring system.

In the near future, we aim to enhance the user interface of the proposed agent-based monitoring system. The present system still lacks some basic functions such as event-driven delivery. Enhancement of the current system is planned, which will incorporate functions to be proposed by GMA.

## Acknowledgement

This work was supported by the Soongsil University Research Fund.

## References

1. I. Foster and C. Kesselman, ed., *The Grid: Blueprint for a New Computing Infrastructure*, Morgan Kaufmann, 1998
2. Foster, I., Kesselman, C., Nick, J. and Tuecke, S. (June 2002). *The Physiology of the Grid: An Open Grid Services Architecture for Distributed Systems Integration*, <http://www.globus.org/research/papers/ogsa.pdf>
3. *Web Services Architecture*, [http://www.w3.org/TR/ws-arch/#service\\_oriented\\_architecture](http://www.w3.org/TR/ws-arch/#service_oriented_architecture)
4. Chen, Anan, Tang, Yazhe, Liu, Yuan, and Li, Ya, *MAGMS: Mobile Agent-Based Grid Monitoring System*, *Lecture Notes in Computer Science*, Vol. 3841, Springer-Verlag
5. Zhang, Zhihuan and Wang, Shuqing, *Agent-Based Framework for Grid Computing*, *Lecture Notes in Computer Science*, Vol. 3032, Springer-Verlag
6. *GMA White Paper*, <http://www.didc.lbl.gov/GGF-PERF/GMA-WG/>
7. *Global Grid Forum (GGF), Grid Monitoring Architecture (GMA) Working Group*, <http://www.didc.lbl.gov/GGF-PERF/GMA-WG/>
8. *Java Agents for Monitoring and Management (JAMM)*, July 2000, <http://www.didc.lbl.gov/JAMM>
9. Cooke, A., Gray, A.J.G., Ma, L., Nutt, W., Magowan, J., Oevers, M., Taylor, P., Byrom, R., Field, L., Hicks, S., Podhorszki, N., Coghlan, B.A., Kenny, S. and O'Callaghan, D., *R-GMA: An Information Integration System for Grid Monitoring*. Robert Meersman, Zahir Tair and Douglass C., Schmidt (eds), *COOPIS 2003, Lecture Notes in Computer Science*, Springer, 2003, Vol. 2888, pp. 462-481
10. *The Datagrid project*. February 2004, <http://eu-datagrid.web.cern.ch/eudatagrid/>
11. *MDS4*, <http://www.globus.org/toolkit/mds/>

# A Peer-to-Peer CF-Recommendation for Ubiquitous Environment

Hyea Kyeong Kim, Kyoung Jun Lee, and Jae Kyeong Kim\*

School of Business Administration, Kyung Hee University  
1, Hoegi-dong, Dongdaemoon-gu, Seoul, 130-701 Korea  
{kimhk, klee, jaek}@khu.ac.kr

**Abstract.** In ubiquitous environment where all entities can freely connect and collaborate with each other from anywhere, the amount of accessible information is overwhelming and desired information often remains unfound. So there is a growing need to provide the personalized recommendation services for the customers in ubiquitous space. This paper suggests a UREC\_P2P(U-RECom-mendation by peer-to-peer), a recommendation procedure in ubiquitous environment adopting P2P technologies combined with collaborative filtering algorithm. UREC\_P2P is implemented and comparatively evaluated with a CF-based recommender system in client-server environment. The evaluation result shows that UREC\_P2P has a good potential to be a preeminent and realistic solution to the recommendation problems encountered in ubiquitous environment.

## 1 Introduction

In ubiquitous computing environment, the amount of accessible information is overwhelming and desired information often remains unfound. In such an environment, there is a need to provide the personalized service suitable to the requirements and the activities of the peers [Chen, 2005; Keegan & O'Hare, 2003; Takemoto et al., 2002]. The CF(Collaborative Filtering)-based recommender system that identifies neighbor peers whose tastes are similar to those of a given peer and recommends contents those neighbor peers have liked in the past [Kim et al., 2005] is known to be a successful solution to provide the personalized service from a huge amount of unnecessary and irrelevant information. However existing CF-based recommender systems are mostly developed for the client-server architecture. The client-server architecture may not be suitable for a ubiquitous environment where all entities can freely build networks on ad hoc basis for collaboration with each other from anywhere. There is no firm distinction between service providers and individual peers, moreover, peers can produce, gather and manage data including off-line shopping related information as well as on-line activity information with personal devices. From these features, peer-to-peer (P2P) technology, which makes it possible to be direct and real-time collaboration between two or more agents running on PCs or handhelds, has been already considered as the alternative architecture of ubiquitous environment [Iwao, 2003; Takemoto et al., 2002].

---

\* Corresponding author.

In this research, we suggest a UREC-P2P(U-RECommendation by peer-to-peer), a recommendation procedure in ubiquitous environment adopting P2P technologies combined with collaborative filtering algorithm, gives support to off-line shoppers with the provision of recommendations in a P2P way. We focus especially on the problem of how agents can find relevant peers depending on what is available from whom without centralized control at the less communication cost.

## 2 UREC\_P2P Service Design

UREC\_P2P model is operated within UrZone, Ubiquitous recommendation service Zone. The UrZone is an extended community as a collection of devices and services that cooperates one another to achieve common goals where the peers are considered to have cooperative relationship among them [You et al., 2005].

The prominent features of UrZone are as follows: Peers are equipped with UPA (Ubiquitous Personal Assistant), a kind of handheld computer. The peers, who have common interest or goals such as searching multimedia contents or shopping for specific items, connect to a certain UrZone on ad hoc basis using UPAs. The UPA hosts a Ubiquitous recommendation Service Agent (UrSA) – an encapsulated autonomous software entity – that is composed of cooperative three modules and charged with the tasks of learning the peer's preference, searching for neighbor peers, and recommending contents or items to the peer.

In UREC\_P2P, the UrSA builds descriptive peer model by compiling own preference of the given peer (called host peer), and then generates a recommendation list appealing to the model. The peer model consists of three parts: *preferred content set*, *neighbor peer set*, and *target peer set*. The preferred content set consists of multiple contents that have saved on a host peer's UPA. Each content is represented as a collection of features that describe its perceptual properties and its centroid point is calculated as its representative [Zhang et. al., 1996]. Each UrSA selects the predefined number of other peers with similar preference as neighbor peer set of the host peer. Whenever a host peer saves a newly obtained contents, the UrSA pushes contents as recommendations to target peer set, is passively organized on the request of other UrSAs. For the purpose of reflecting host peer's most recent preference, the peer model of UREC\_P2P is dynamically updated whenever a host peer connects the UrZone.

## 3 UREC\_P2P Procedure

UREC\_P2P procedure is done with iterative three distinct modules, *an Event-driven push module*, *a Top-k filtering module*, and *a Neighbor formation module*.

**Event-driven push module.** UREC\_P2P generates recommendations with an Event-driven Push strategy. When-ever a peer saves a content on UPA, the newly saved content is updated to the pre-ferred content set and then forwarded to the target peers in real time. The push way recommendation can particularly emphasize the most recent ratings of neighbor peers, which leads to faster spread of newly obtained

contents. This is significantly different from traditional CF techniques that suffer from a new content problem, which make it difficult to recommend a newly released content until some ratings of the contents become available.

**Top-k filtering module.** When a host peer connects the UREC\_P2P network, the top-k filtering module of UrSA retrieves  $k$  recommended contents among the pushed contents. The contents pushed by neighbor peers have been sequentially accumulated in the queue until the queue is full with  $C$  contents, after that, the accumulated contents are discarded orderly from the beginning. Therefore a queue keeps on maintaining recently received  $C$  contents. The recommendation list is calculated for the ratings of contents in a queue only. That means very past ratings are discarded to place particular emphasis on popular contents among the neighbor peers at that time. Most existing CF-based recommender systems generally use all of the peer ratings in the past. The module makes recommendation list using *PLS*, purchase likeliness score [Kim et al., 2005]. The top- $k$  contents are presented to the host peer  $h$  and the peer skims through the list to see if there are any contents of interest. Then, the peer may save the preferred contents on his/her UPA.

**Neighbor formation module.** Each UrSA generates a neighbor peer set based on the similarity between preferred content sets of the host peer and other peers using centroid Euclidian distance function [Zhang et Al., 1996]. The module keeps neighbor peer set as the peers with consistently similar preference to a host peer through the dynamic neighbor re-formation. When the preference of a current neighbor peer becomes different from that of the host peer, the neighbor peer is excluded from the neighbor peer set. For the replacement of the excluded neighbor, the host peer explores more similar peers within the range of the neighbor peer set of the most similar neighbor peer. The intuition behind the range of limited exploration is that neighbor peers with highly similar tastes could offer much chance of useful information to the host peer. It places a limit on the depth of search for more relevant neighbor peers, which leads to shorten the paths for pushed recommendations and hence improve the system performance with hindrance to communicate with distant peers.

## 4 Experimental Evaluation

### 4.1 Experimental Design

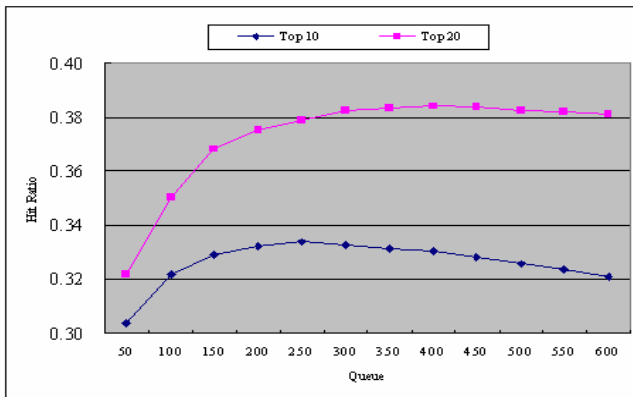
We compare the performance of a UREC\_P2P with that of a centralized benchmark recommendation methodology. The centralized benchmark system does not perform the recent rating-based filtering and local information-based neighbor reformation.

We use real transaction and content data in mobile commerce offering character images provided from a leading content provider in Korea. The data contain 8,776 images, 1,921 customers, and their 55,321 transactions during the period between June 1, 2004 and August 31, 2004. To characterize images, HSV (i.e. hue, saturation, and value of color) based color moment was selected as visual features [Porkaew et al., 1999]. The transaction data during the three months are experimentally divided

into two sets, a training set and a test set to design a noble experiment method. Host peers are determined as the users who have purchased at least one image during the training period and initial preferred content set and neighbor peer set of each host peer is generated from transaction records for the training period. And each peer receives recommendations at each connection date for the test period, and then we observe whether the recommendations match the real purchase of each host peer or not. Two metrics, *hit ratio* and *response time* are employed for evaluation of accuracy and performance respectively [Kim et al. 2005].

## 4.2 Results and Discussion

We performed experiments where we varied the queue size from 50 to 600 with an increment 50. From the experiment, we make an important observation that the quality of recommendation improves as the queue size increases, but after a certain level the improvement slows down and eventually the recommendation quality becomes worse over all neighbor peer sizes as shown in Fig.1.



**Fig. 1.** Impact of queue size

This indicates that the excessive queue size causes violation of reflecting the current preference, which leads to lower quality of recommendations. It confirmed that the recent rating-based filtering of UREC\_P2P is a reasonable suggestion to enhance the quality of recommendations especially in the domains significantly affected by newly released contents.

To compare with the centralized benchmark system, the experiments were carried out with varied number of neighbors at  $k=20$ , and computed the corresponding hit ratio and response time. In this evaluation, UREC\_P2P is tuned with the ideal levels of those queue sizes to make fair comparisons. Fig. 2 shows that UREC\_P2P works dramatically better than the benchmark system at all number of neighbor sizes. From the results of hit ratio, UREC\_P2P gains improvement of about 4.7 times on the average.

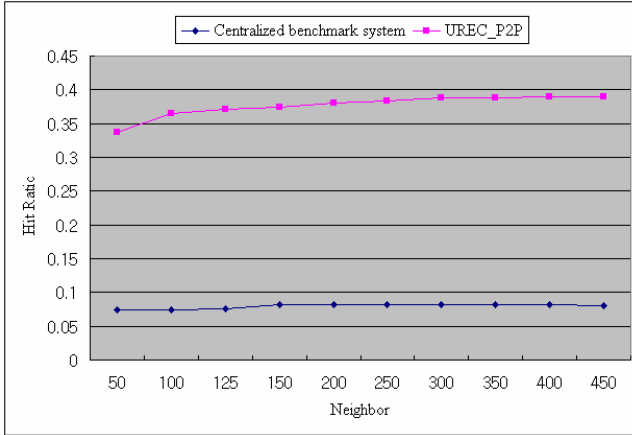


Fig. 2. Quality comparison with benchmark system at  $k=20$

To evaluate the rates of improvement of UREC\_P2P over that of the benchmark system, we measure response time of two systems. Table 1 also shows the response time of each top 20 recommendation provided by two systems. Looking into the results, we can see that the response time of UREC\_P2P is about 176 times faster than that of the benchmark system.

Table 1. Performance comparison

	UREC_P2P	Benchmark system
Response time (sec.)	0.009	1.9541

## 5 Conclusion

In ubiquitous computing environment, all entities can freely connect and collaborate with each other from anywhere. Therefore, the distributed computing and ad hoc nature of the P2P system becomes main features of ubiquitous environment. This research suggests a UREC\_P2P, a recommendation procedure in ubiquitous environment, adopting P2P technologies combined with collaborative filtering algorithm.

UREC\_P2P has the following characteristics; (1) UREC\_P2P deals with the most current preference of peers using queue and learns peer preference in real time from peer’s content selections without requiring peer’s explicit ratings. (3) A peer’s event, such as saving contents, triggers recommendations with push way without centralized control for the purpose of faster distribution of contents. (4) Similar neighbor peers are dynamically selected from neighbor peer’s neighbors only. These characteristics of UREC\_P2P make each peer keep better constituent neighbors and can obtain more current preference related recommendations.

Our experiment shows that UREC\_P2P offers remarkably higher quality of recommendations than the centralized benchmark recommender system, which results

from UREC\_P2P's accelerated reflecting of the changes of peer preferences. Moreover UREC\_P2P works dramatically faster than the centralized system, which results from a recent rating-based filtering and local information-based neighbor re-formation. These results are very important in ubiquitous computing environment, because the number of contents and peers grow very fast, and UPA may have inherently a limitation of computing power. These experimental results prove that UREC\_P2P has good potential to be a realistic solution to the recommendation problems encountered in multimedia content sharing in ubiquitous computing environment.

The UrZone of this paper is restricted to cyber market dealing with image contents only. However, UREC\_P2P has flexibility to share any contents or product information, so it will be a promising future research area, the adaptation of UREC\_P2P for diverse UrZones. Furthermore, it is required to develop an experimental design for the evaluation of UREC\_P2P using data from ubiquitous computing environment.

## References

1. Chen, A.: Context-Aware Collaborative Filtering System: Predicting the User's Preference in Ubiquitous Computing. *LoCA 2005, LNCS 3479 (2005)* 244-253.
2. Iwao, T., Amamiya, S., Zhong, G., and Amamiya, M.: Ubiquitous Computing with Service Adaptation Using Peer-to-Peer Communication Framework. In *9<sup>th</sup> IEEE Workshop on Future Trends of Distributed Computing Systems (2003)* p. 240
3. Keegan, S. and O'Hare, G.: EasiShop: Enabling uCommerce through Intelligent Mobile Agent Technologies. *LNCS 2881 (2003)* 200-209.
4. Kim, H., Kim, J., Cho, Y.: A Collaborative Filtering Recommendation Methodology for Peer-to-Peer Systems, *EC-Web 2005, LNCS 3590 (2005)*, pp. 98–107.
5. Porkaew, K., Chakrabarti, K. Mehrotra, S.: Query Refinement for Multimedia Similarity Retrieval in MARS, In *Proc. Of the 7th ACM Multimedia Conference. (1999)* 235-238.
6. Takemoto, M., Sunage, H., Tanaka, K., Matsumura, H., and Shinohara, E.: The Ubiquitous Service-Oriented Network (USON) – An Approach for a Ubiquitous World based on P2P Technology. In *Proc. 2<sup>nd</sup> International Conference on Peer-to-Peer Computing. (2002)*
7. You, S., Choi, J., Heo, G., Choi, D., Park, H., Kim, H., Cho, W.: COCOLAB: Supporting Human Life in Ubiquitous Environment by Community Computing. In *1<sup>th</sup> Korea/Japan Joint Workshop on Ubiquitous Computing & Networking Systems. (2005)*.
8. Zhang, T., Ramakrishnan, R., Livny, M.: BIRCH: An Efficient Data Clustering Method for Very Large Database, In *Proc. of the 1996 ACM SIGMOD International Conference on Management of Data. (1996)*, 103 – 114.

# Platform-Level Multiple Sensors Simulation Based on Multi-agent Interactions

Xiong Li, Kai Wang, Xianggang Liu, Jiuting Duo, and Zhiming Dong

Department of Command and Administration, Academy of Armored Force Engineering,  
100072 Beijing, China  
lixiong2609@126.com

**Abstract.** In order to support the advanced concept technology demonstration of intelligence reconnaissance activities, multiple sensors simulation needs to provide enough detail to examine sensing dynamics on future battlefield. However, most current researches lack a modeling mechanism based on platform-level agent interactions and limit the implementation of agent-based simulations. In this paper, platform-level agent interactions based modeling and simulation technology is proposed to solve the problem. According to agent definition of mapping function from perception sequences to actions, the mapping from multiple sensors system's members to respective agents is set up. Thus, the sensor entity agent model is designed. Moreover, sensor agent interactions model is presented to support multiple sensors simulation based on multi-agent interactions by using an improved Contract Net Protocol, where state transitions and augmented transition network are studied to represent the interactions. The established demonstration system proves the feasibility and efficiency of our model.

**Keywords:** agent, multi-agent system, interactions, simulation.

## 1 Introduction

Agents and multi-agent systems have turned out to be useful for a wide range of application domains where difficult problems have to be dealt with [1]~[7]. Agent-based simulation approaches to military simulation field gained increasing attention in recent years [5]~[7]. However most existent models and systems can not provide enough detail to examine important dynamics in tactical warfare process, e.g., unpredictability of military system operations and entity interactions. They can only perform unit-level simulation. Moreover, almost all researches on agent-based simulation are not based on multi-agent interactions. Thus there are a lot of difficulties when the systems are implemented, since agents and multi-agent systems are complex and have many properties such as autonomy, reactivity, sociality, adaptability and intelligence. It is impossible to take all these factors into account.

Multiple sensors (including photo-reconnaissance vehicles, radar reconnaissance vehicles, armored reconnaissance vehicles and information processing vehicles) have administrative levels and interactions, e.g., sending or receiving reconnaissance



orders. Since multiple sensors system is in substance a complex distributed artificial intelligence system, platform-level agent-based simulation method is applicable. Meantime, because of the complexity, it is sensible to center on these entity agents interactions and ignore other peculiarities to lighten burden. In this paper, we design a platform-level multiple sensors simulation system model based on multi-agent interactions to build underlying mechanisms for the advanced concept technology demonstration of intelligence reconnaissance activities on future battlefield.

## 2 Mapping from Sensors to Agents

An agent may have beliefs, desires, intentions, and it may adopt a role or have relationships with others. Multiple sensors system is so alike a distributed multi-agent system in behaviors that we can set up a mapping from its internal members, i.e. platform-level sensor entities, to entity agents, e.g., armored reconnaissance vehicle → armored reconnaissance vehicle agent.

Of course, the established sensor entity agent model must be as simple as possible, but it can exhibit complex behavior patterns and provide valuable information about the dynamics of the real world, i.e., sensing action on distributed battlefield. These agents are often required to attain goals that are only possible, made easier or satisfied more completely by interacting with other agents. In this context, “interaction” is used as a generic term for activities such as cooperation (working together to achieve a common objective), coordination (arranging inter-dependent activities) and negotiation (coming to a mutually acceptable agreement on some matter).

```

function SKELETON-AGENT(perception) return action
static: memory, the agent's memory of the world
memory ← UPDATE-MEMORY(memory, perception)
action ← CHOOSE-BEST-ACTION(memory)
memory ← UPDATE-MEMORY(memory, action)
return action

```

**Fig. 1.** A skeleton sensor entity agent

In our model all sensor entity agents have the same skeleton (See Fig. 1), namely, accepting perception from an environment and generating actions. On each invocation, the sensor agent's memory is updated to reflect the new perception, the best action is chosen, and the fact that the action was taken is also stored in memory. The memory persists from one invocation to the next.

## 3 Interactions Model

In this paper we use an improved Contract Net Protocol (CNP) [1], [3] which has constraints and confirmation functions. The Contract-Net Initiator as a manager represents the information processing vehicle, and all other Participants as contractors represent the other reconnaissance vehicle agents. Of course, the roles of Initiator and Participants are changed once interaction relation changes.

In our model the Initiator wishes a task to be performed by one or a group of agents according to some arbitrary function which characterizes the task. The Initiator issues the call for proposals, and other interested agents can send proposals. In contrast to original CNP, there is no need to do anything if an agent playing a role of a potential Participant is not interested in submitting proposals. That means that our CNP model from the very beginning relies on the notion of timeout, i.e. some actions need to be performed in the event of a lack of enough proposals or even in the case of a complete lack of proposals.

The proposals are collected by the Initiator, and then they are refused or accepted. The accepted proposals can be cancelled, either, by the Initiator via a cancel action, or by the Participant via a failure action. In case of cancellation other submitted proposals can be reconsidered, or a completely new call for proposals can be issued. The Initiator’s and Participant’s possible output of messages and an example rule in the request protocol are presented in Fig. 2.

<pre> Process Initiator := Send !msg(id, add(p, nil), cnt(request)) []Send !msg(id, add(p, nil), cnt(cancel)) endproc process Responder := Send !msg(id, add(ir, nil), cnt(not-understood)); []Send !msg(id, add(ir, nil), cnt(reject)); []Send !msg(id, add(ir, nil), cnt(agree)); []Send !msg(id, add(ir, nil), cnt(failure)); []Send !msg(id, add(ir, nil), cnt(inform-done)); []Send !msg(id, add(ir, nil), cnt(inform-result)); endproc                 </pre>	<pre> [messageType eq request] → ( Send !msg(...not-understood...); [] Send !msg(...reject...); [] Send !msg(...agree...); )                 </pre>
---	---

**Fig. 2.** Possible output of messages and example rule in the request protocol

The consideration for the required interactive rules includes the relationships between actions. For example, some requests are allowed to delay, while some others have to be done immediately. To express that two actions occur one after another, we can combine these two actions using the action prefix operator (;). Otherwise, we should assemble them in parallel to allow them and other actions to occur alternately. In the above example rule, when a Participant receives a request, it may respond with not-understood, reject or agree.

<pre> initial, started, terminal → InitiatorSessionState                 </pre>	<pre> [st eq started] → ( Send !msg(...not-understood...); []Send !msg(...failure...); []Send !msg(...inform-done...); []Send !msg(...inform-result...); )                 </pre>
---	---

**Fig. 3.** Initiator role’s state and messages send out by a Participant

Then, we can identify the sensor entity agent states from an agent’s standpoint. States identification can be done incrementally. We can add a new state whenever we find it necessary during the development. For example, the Initiator role’s state and

messages send out by a Participant are shown in Fig. 3. In its initial state, the Initiator can send out a request; in its started state, it can send out a cancel message; and in its terminal state, it can do nothing related to this interaction. After a Participant accepts a request, it can send out the messages at any time.

In our improved CNP, the interaction is started by the information processing vehicle agent who acts as a Selector or Initiator issuing a call for proposals, e.g. scouting the No. 1 target in 1283 highland. These reconnaissance vehicle agents who act as Participants or potential Contractors respond with proposals, which the information processing vehicle agent either rejected or accepted. Accepted proposals can be either cancelled by the information processing vehicle agent, or executed by a certain reconnaissance vehicle agent, who later informs the information processing vehicle agent of success or failure of the execution. In this process, there are iterative confirmations to ensure performing reconnaissance tasks.

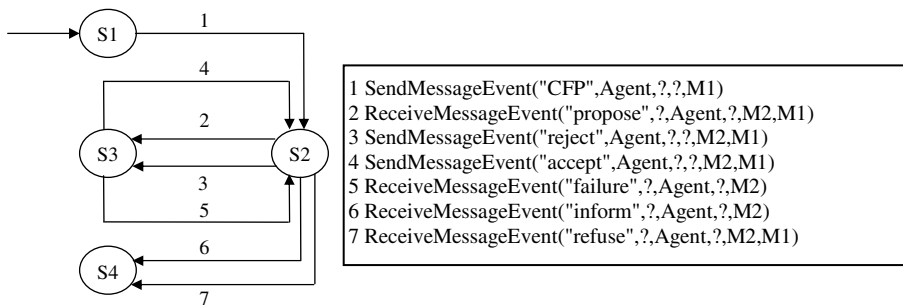


Fig. 4. Machine state for the Initiator

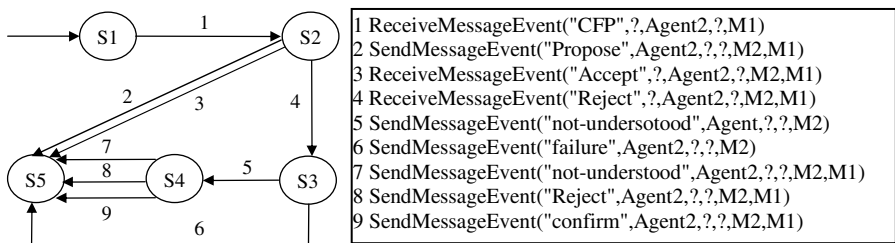


Fig. 5. Machine state for the Participant

In order to represent distinctly the interaction pattern of sensor entity agents, we apply state machines approach. Thus, each agent interaction model is represented by an augmented transition network. A transition represents an interaction event (sending or receiving a message), i.e., interaction events represent the exchanged messages. We distinguish two kinds of interaction events: ReceiveMessage and SendMessage. The attributes of the SendMessage and ReceiveMessage interaction events are similar to the attributes of agent communication language (ACL) messages:

- SendMessage (Communicative act, sender, receiver, content, reply-with, ...).
- ReceiveMessage (Communicative act, sender, receiver, content, reply-with, ...).

We introduce the “wild card” character (?) to filter various messages. For example, in the interaction event ReceiveMessage (“CFP”, “X”, “Y”, ?), the content is unconstrained. So, this interaction event can match any other interaction event with the communication act CFP, the sender “X”, the receiver “Y” and any content.

Fig. 4 and Fig. 5 show examples of augmented transition network that represent the interactions model of the roles Initiator and Participant described above.

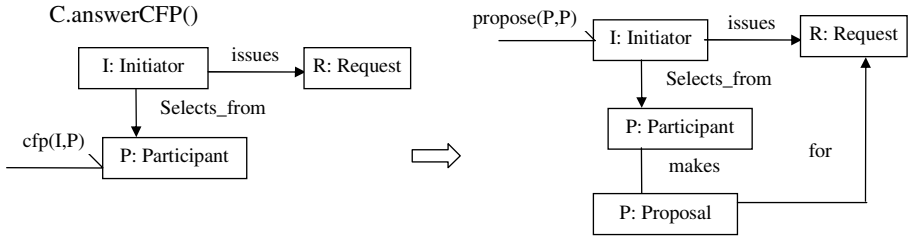


Fig. 6. The generic answerCFP rule

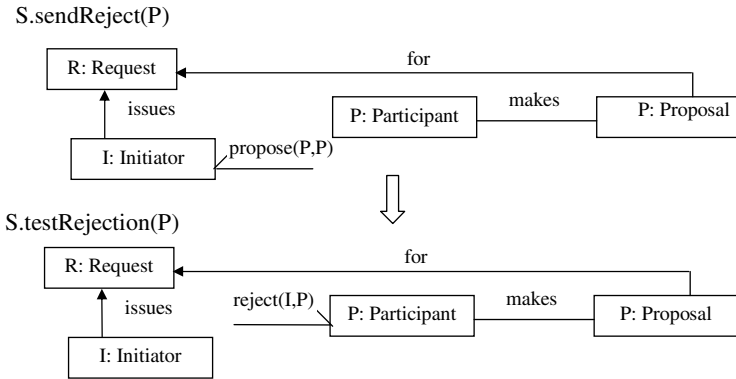


Fig. 7. The generic sendReject rule

In Fig. 6 and Fig. 7, graph transformation rules for the operations of the generic role classes are specified [4]. A rule consists of a left-hand and right-hand side describing the precondition and the effect of the operation, respectively. The precondition describes when the rule is applicable. For example, the answerCFP rule is only applicable if the corresponding structure exists (i.e. there is an Initiator or Selector instance connected to a Participant instance by a selects-from link and to a Request instance by an issues link) and the CFP-message has been sent to the Participant or Contractor instance. The right-hand side of the rule describes the structure resulting from the operation. In the case of the answerCFP rule, a new Proposal instance is created and a propose message is sent to the Initiator.

The demonstration system that we establish can be illustrated by Fig. 8. Fig. 8 presents partial, dynamic and real-time two-dimension battlefield situation information during platform-level multiple sensors simulation based on multi-agent

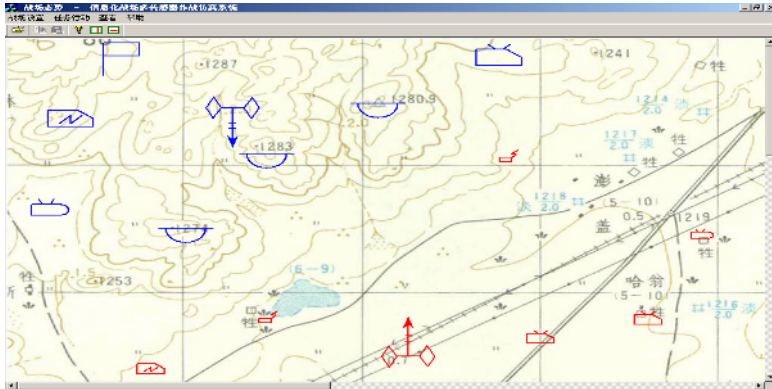


Fig. 8. Demonstration system

interactions. When we run the system, we can obtain some results according with real situation, and find that these sensor agents performed successfully the task of intelligence reconnaissance on tactical virtual battlefield by the improved CNP.

## 4 Conclusion

In this paper, a multi-agent platform-level multiple sensors entities simulation model is studied. By setting up the mapping from multiple sensors to respective entity agents, we put forward the sensor agent model and the multi-agent system model. In order to support simulation based on multi-agent interactions, we presented an improved CNP with constraints and confirmation functions. We also studied its state transitions and augmented transition network to represent the Initiator and Participant roles interactions. The demonstration system shows that our model can be used to realize the dynamic platform-level battlefield reconnaissance simulation.

## References

1. Zhongzhi Shi: Intelligent Agents and Their Applications. Beijing: Science Press (2000)
2. V. Lesser: Autonomous Agents and Multi-Agent Systems. Kluwer (1998)
3. Haque, N. R. Jennings, L. Moreau: Resource Allocation in Communication Networks Using Market-Based Agents. In: Proc. 24th Int. Conf. on Innovative Techniques and Applications of AI (2004) 187–200
4. Ralph Depke, Reiko H., Jochen M. K.: Roles in Agent Oriented Modeling. International Journal of Software Engineering and Knowledge Engineering. 3(2001) 281–302
5. Xiong Li, Xiaobin Liu, Na Hao: Multi-agent-oriented Modeling for Intelligence Reconnaissance System. In: Hong Shen, Koji Nakano (eds.): Parallel and Distributed Computing. IEEE, Los Alamitos, California (2005) 563–566
6. Xiong Li, Degang Liu, Hua Cong: Multi-Agent-Based Space Information Interaction Simulation Model. In: Shan Zhong, Xiulin Hu (eds.): Proc. Int. Conf. on Space Information Technology, SPIE Press (2005) 598509-1–598509-5
7. Jeffrey R. Cares: The Use of Agent-Based Models in Military Concept Development. In: Proc. Int. Conf. on 2002 Winter Simulation Conference (2002) 935–939

# Modeling Viral Agents and Their Dynamics with Persistent Turing Machines and Cellular Automata

Jingbo Hao, Jianping Yin, and Boyun Zhang

School of Computer Science, National University of Defense Technology,  
Changsha 410073, China  
{hjb, jpyin}@nudt.edu.cn, hnjxzby@yahoo.com.cn

**Abstract.** A computer virus is a program that can generate possibly evolved copies of itself when it runs on a computer utilizing the machine's resources, and by some means each copy may be propagated to another computer in which the copy will have a chance to get executed. And we call a virus instance as a viral agent since it is autonomous during its execution by choosing what action to perform in the computer without a user's intervention. In the paper we develop a computational model of viral agents based on the persistent Turing machine (PTM) model which is a canonical model for sequential interaction. The model reveals the most essential infection property of computer viruses well and overcomes the inherent deficiency of Turing machine (TM) virus models in expressing interaction. Then on that basis we deduce several helpful theorems about viral agents. Finally we also discuss modeling of viral agent dynamics with cellular automata (CAs) and get some useful results.

**Keywords:** Viral Agent, Persistent Turing Machine, Dynamics, Cellular Automaton.

## 1 Introduction

A computer virus is a program that can generate possibly evolved copies of itself when it runs on a computer utilizing the machine's resources, and by some means each copy may be propagated to another computer in which the copy will have a chance to get executed. And we call a virus instance as a viral agent since it is autonomous during its execution by choosing what action to perform in the computer without a user's intervention. Computer viruses are always a serious threat against information security. Although various countermeasures have hitherto been adopted, computer viruses can not be prevented radically.

To understand computer viruses in essence, a proper computational models of viral agents is certainly necessary. There have been a few such models [1–3] which leave out an important property of viral agents – interaction. This is due to the inherent limitations of the classical computational models, such as the most typical Turing machines (TMs). A TM can only model a function-based transformation of an input to an output. When the area of computation is extended from algorithms to processes, TMs are no longer appropriate to capture all the features of computation including

interaction. A computing agent has the interaction property if it has input and output actions that interact with an external environment not under its control [4]. Persistent Turing machines (PTMs) are a canonical model for sequential interaction, and the concepts of PTMs have been well defined in [5]. In general a viral agent interacts with its environment sequentially because no viral agent works like an engaged web server, and so PTMs are quite suitable for viral agent modeling.

Furthermore, correlated viral agents will form a multi-agent system (MAS) which can exhibit some kinds of collective dynamics, especially for propagation. Cellular automata (CAs) are discrete dynamical systems whose individual components are rather simple, yet that can exhibit highly complex and unpredictable behavior due to these simple components' mutual interaction and synergy [6]. Brooks et al. have presented a model of network dynamics based on CAs [7], and so we can similarly model viral agent dynamics with CAs.

The rest of the paper is organized as follows. In section 2 we introduce the basic concepts of PTMs. In section 3 we develop a computational model of viral agents based on PTMs. In section 4 we discuss modeling of viral agent dynamics with CAs. Finally we draw the conclusion.

## 2 Concepts of PTMs

A PTM is a nondeterministic 3-tape TM (N3TM) with a read-only input tape, a read/write work tape, and a write-only output tape [5]. Upon receiving an input token from its environment on its input tape, a PTM computes for a while and then outputs the result to the environment on its output tape, and this process is repeated forever. A PTM performs persistent computations in the sense that the work tape contents are maintained from one computation step to the next, where each PTM computation step represents an N3TM computation.

**Definition 2.1.** An N3TM is a quadruple  $\langle K, \Sigma, \delta, s_0 \rangle$ , where:

- $K$  is a finite set of states.
- $\Sigma$  is a finite alphabet containing the blank symbol #, but not containing L (left) and R (right).
- $\delta \subseteq K \times \Sigma \times \Sigma \times \Sigma \times (K \cup \{h\}) \times (\Sigma \cup \{L, R\}) \times (\Sigma \cup \{L, R\}) \times (\Sigma \cup \{L, R\})$  is the transition relation.
- $s_0 \in K$  is the initial state.
- $h \notin K$  is the halting state.

**Definition 2.2.** A PTM is N3TM having a read-only input tape, a read/write work tape, and a write-only output tape.

**Definition 2.3.** Let  $M$  be a PTM having alphabet  $\Sigma$ , and let  $w_i$ ,  $w$ ,  $w'$  and  $w_o$  be words over  $\Sigma$ . We say that  $w \xrightarrow[M]{w_i / w_o} w'$  (yields in one macrostep) if  $M$ , when started in its initial control state with its heads at the beginning of its input, work, and output tapes containing  $w_i$ ,  $w$ , and  $\epsilon$  respectively, has a halting computation that produces  $w_i$ ,  $w'$  and  $w_o$  as the respective contents of its input, work, and output tapes.

Should M’s computation diverge, we write  $w \frac{w_i / \mu}{M} > s_{div}$ , where  $s_{div}, \mu \notin \Sigma$  and  $s_{div}$  is a special “divergence state” such that  $s_{div} \frac{w_i / \mu}{M} > s_{div}$  for all inputs  $w_i$ ; and  $\mu$  is a special output symbol signifying divergence.

**Definition 2.4.** Let M be a PTM with alphabet  $\Sigma$ , then reach(M), the reachable states of M, is defined as:

$$\begin{aligned} & \{ \varepsilon \} \cup \{ w \in \Sigma^* \cup \{s_{div}\} \mid \exists k \geq 1, \exists w^1_i, \dots, w^k_i \in \Sigma^*, \\ & \exists w^1_o, \dots, w^k_o \in \Sigma^* \cup \{\mu\}, \exists w^1, \dots, w^k \in \Sigma^* \cup \{s_{div}\} : \\ & \varepsilon \frac{w^1_i / w^1_o}{M} > w^1, w^1 \frac{w^2_i / w^2_o}{M} > w^2, \dots, w^{k-1} \frac{w^k_i / w^k_o}{M} > w^k \wedge w = w^k \}. \end{aligned}$$

**Definition 2.5.** Let U be a PTM with alphabet  $\Sigma_U$ , let M be a PTM with alphabet  $\Sigma_M$ , let  $w_i, w_o, w$  and  $w'$  be strings over  $\Sigma_M$ , and let  $\eta : M, \Sigma_M^* \rightarrow \Sigma_U^*$  be a one-to-one encoding function for the transition relation and alphabet of M. Then, U is a universal PTM simulating M if:

- U has an initial macrostep  $\varepsilon \frac{\langle \eta(M), \eta(w) \rangle}{U} > \langle \eta(M), \eta(w) \rangle$ .
- If M has a halting computation  $w \frac{w_i / w_o}{M} > w'$ , the U has a halting computation  $\langle \eta(M), \eta(w) \rangle \frac{\eta(w_i) / \eta(w_o)}{U} > \langle \eta(M), \eta(w') \rangle$ .
- If M diverges, written  $w \frac{w_i / \mu}{M} > s_{div}$ , then U diverges, written  $\langle \eta(M), \eta(w) \rangle \frac{\eta(w_i) / \mu}{U} > s_{div}$ .
- If  $s_{div} \frac{w_i / \mu}{M} > s_{div}$ , then  $s_{div} \frac{\eta(w_i) / \mu}{U} > s_{div}$ .

**Theorem 2.1.** There is a PTM U that is a universal PTM. (Refer to [5] for the proof)

### 3 Modeling Viral Agents with PTMs

Modern general-purpose computers are implemented in accordance with the idea of universal TMs. However, as the theoretical basis of computers universal TMs are unable to describe a continuous computational process of a computer during which unpredictable interaction events sequentially happen. It is conceivable that universal PTMs can be used to make up the deficiency since a universal PTM can simulate a computational process of an arbitrary PTM which can represent a program properly. Upon that we model viral agents with PTMs based on our previous work in [8].

**Definition 3.1.** For a given universal PTM U, the set of all its programs  $TP = \{ \eta(M) \mid M \text{ is a PTM} \}$ .



**Definition 3.2.** For a given universal PTM  $U$ ,  $V$  is a program set iff  $V \subseteq TP$  and  $V \neq \emptyset$ .

**Definition 3.3.** For a given universal PTM  $U$ , the set of all the program sets  $TS = \{V \mid V \subseteq TP \text{ and } V \neq \emptyset\}$ .

**Definition 3.4.** For a given universal PTM  $U$ , for  $V \in TS$  and  $v = \eta(M) \in V$ ,  $v \xRightarrow{U} V$  iff  $\exists w \exists w' \exists w_i \exists v' : w, w' \in reach(M), w_i \in \Sigma^*_M, v' \in V :$   
 $\langle v, \eta(w) \rangle \xrightarrow{U} \frac{\eta(w_i) / v'}{U} \times \langle v, \eta(w') \rangle .$

**Definition 3.5.** With respect to all kinds of universal PTMs, the whole viral set  $WS = \{(U, V) \mid U \text{ is a universal PTM, } V \in TS \text{ for } U, \text{ and } \forall v \in V, v \xRightarrow{U} V\}$ .

**Definition 3.6.**  $V$  is a viral set with respect to a universal PTM  $U$  iff  $(U, V) \in WS$ .

**Definition 3.7.**  $v$  is a viral agent with respect to a universal PTM  $U$  iff  $v \in V$  and  $(U, V) \in WS$ .

**Definition 3.8.** A smallest viral set with respect to a given universal PTM is a viral set of which no proper subset is a viral set.

These definitions recursively model a viral agent as a PTM which can generate possibly evolved isogenous PTMs through interaction with the environment. In this way the model adequately exhibits the infection essence of a viral agent. Next we deduce several helpful theorems about viral agents.

**Theorem 3.1.** For any finite number  $n$  and an arbitrary universal PTM  $U$ , there exists a smallest viral set with  $n$  elements.

**Proof.** For any finite number  $n$  and an arbitrary universal PTM  $U$ , we can construct  $n$  PTMs  $M_1, \dots, M_n$ , so that:

$$\forall w_{i1} \in \Sigma^*_{M_1}, \dots, \forall w_{in} \in \Sigma^*_{M_n}, \forall w_1 \in reach(M_1), \dots, \forall w_n \in reach(M_n),$$

$$\exists w'_1 \in reach(M_1), \dots, \exists w'_n \in reach(M_n) :$$

$$\langle \eta(M_1), \eta(w_1) \rangle \xrightarrow{U} \frac{w_{i1} / \eta(M_2)}{U} \times \langle \eta(M_1), \eta(w'_1) \rangle, \dots,$$

$$\langle \eta(M_{n-1}), \eta(w_{n-1}) \rangle \xrightarrow{U} \frac{\eta(w_{in-1}) / \eta(M_n)}{U} \times \langle \eta(M_{n-1}), \eta(w_{n-1}') \rangle,$$

$$\langle \eta(M_n), \eta(w_n) \rangle \xrightarrow{U} \frac{\eta(w_{in}) / \eta(M_1)}{U} \times \langle \eta(M_n), \eta(w_n') \rangle .$$

Therefore  $\forall v \in \{\eta(M_1), \dots, \eta(M_n)\} = V, v \xRightarrow{U} V, (U, V) \in WS$  and no proper subset of  $V$  can be a viral set, and so  $V$  is a smallest viral set with  $n$  elements.

**Theorem 3.2.** Any union of viral sets is also a viral set, i.e., if  $(U, V_1) \in WS$  and  $(U, V_2) \in WS$  then  $(U, V_1 \cup V_2) \in WS$ .

**Proof.**  $\forall v \in V_1 \cup V_2,$

if  $v \in V_1,$  since  $(U, V_1) \in WS,$  then  $v \xRightarrow{U} V_1,$  so  $v \xRightarrow{U} V_1 \cup V_2;$

if  $v \in V_2,$  since  $(U, V_2) \in WS,$  then  $v \xRightarrow{U} V_2,$  so  $v \xRightarrow{U} V_1 \cup V_2.$

Therefore  $\forall v \in V_1 \cup V_2, v \xRightarrow{U} V_1 \cup V_2,$  and so  $(U, V_1 \cup V_2) \in WS.$

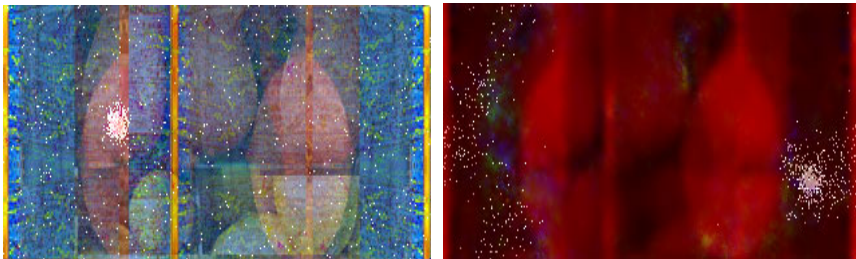
**Theorem 3.3.** The whole viral set WS is undecidable.

**Proof.** Each macrostep of a PTM is a complete Turing computation and PTMs are an extension to TMs [5], so any TM can be implemented by a certain PTM. If the whole viral set WS is decidable, we can deduce that the whole viral set defined in [8] is also decidable which however has been proven to be false.

### 4 Modeling Viral Agent Dynamics with CAs

In a classical CA, time is discrete while the space is divided into an n-dimensional lattice of cells, each cell representing an identical finite state machine (FSM). All cells change their states simultaneously with each using the same update function to determine its next state according to its current state and the current states of its neighboring cells. This set of adjacent cells is called a neighborhood, and the center cell is included in its own neighborhood by convention. There are many variations of the classical CA. For example, each cell in a CA can have its own state set (state-heterogenous) or update function (rule-heterogenous). Brooks et al. used a rule-heterogenous CA as a generalized networking substrate and modeled several mobile code paradigms [7]. They abstracted away a more complex network topology in favor of a lattice topology in which network nodes are connected to their closest neighbors. This work focuses on modeling network flows and is helpful to the study of network congestion. So we can use the method to research into network flow and network congestion caused by viral agents. However, if we care more about virus propagation trends, a more realistic model is needed.

Tosic introduced two classes of discrete dynamical system models that result in the parallel and sequential CA models once some heterogeneity is allowed in modeling and analysis of large-scale MASs [6]. The two models are respectively sequential dynamical systems (SDSs) and synchronous dynamical systems (SyDSs). These two



**Fig. 1.** Viral agent dynamics in an image (left at the start point and right at the end point) [9]

variations of the classical CA (or graph automata) are obviously more precise than the network model in [7]. We can use these models to simulate virus propagation more realistically, which will be done in our subsequent work.

In Nechvatal's Computer Virus Project 2.0 [9], computer viruses are modeled to be autonomous agents living in an image. The behavior of a viral agent is modeled as a looping activity in which the agent will pick up information from its environment, decide on a course of action, and carry it out. As the viral agent executes, it moves to one of the adjacent squares and changes the current pixel. It can even reproduce itself or die (see Fig. 1). The algorithm idea adopted by the project is very similar to CAs, which proves that CAs can be used for faithful modeling of viral agent dynamics.

## 5 Conclusion

In the paper we develop a computational model of viral agents based on PTMs which are a canonical model for sequential interaction. The model reveals the most essential infection property of computer viruses well and overcomes the inherent deficiency of TM virus models in expressing interaction. On that basis we deduce several helpful theorems about viral agents. We also discuss modeling of viral agent dynamics with CAs, and get some useful results. But due to the limitation of time, we still have many problems left to solve which will be done in our subsequent work.

**Acknowledgments.** This work has been sponsored by the National Natural Science Foundation of China (project no. 60373023).

## References

1. Cohen, F.: Computational Aspects of Computer Viruses. *Computers & Security*, Vol. 8, No. 4 (1989) 325–344
2. Adleman, L.M.: An Abstract Theory of Computer Viruses. In: Goldwasser, S. (ed.): *Advances in Cryptology – CRYPTO'88*, LNCS 403, Springer-Verlag Berlin Heidelberg (1990) 354–374
3. Leitold, F.: Mathematical Model of Computer Viruses. In: *EICAR 2000 Best Paper Proceedings*, Brussels, Belgium (2000) 194–217
4. Wegner, P.: Towards Empirical Computer Science. *The Monist*, Vol. 82, No. 1 (1999) 58–108
5. Goldin, D., Smolka, S., Attie, P., Sonderegger, E.: Turing Machines, Transition Systems, and Interaction. *Information and Computation*, Vol. 194, No. 2 (2004) 101–128
6. Tasic, P.T.: On Modeling and Analyzing Sparsely Networked Large-Scale Multi-Agent Systems with Cellular and Graph Automata. To appear in: *Proceedings of the 2nd Workshop on Modeling of Complex Systems by Cellular Automata*, Reading, UK (2006)
7. Brooks, R., Orr, N., Zachary, J., Griffin, C.: An Interacting Automata Model for Network Protection. In: *Proceedings of the 5th International Conference on Information Fusion*, Annapolis, USA (2002) 1090–1097
8. Hao, J., Yin, J., Zhang, B.: Proof of the Virus Decidability Theorem Based on a Universal Turing Machine Model (in Chinese). In: *Proceedings of the National Annual Conference on Theoretical Computer Science*, Qinhuangdao, China (2005) 243–245
9. Nechvatal, J., Sikora, S.: Red Ovoid Attack - Computer Virus Project 2.0. (2001) <http://www.computerfinearts.com/collection/nechvatal/redattack/>

# A Lightweight Architecture to Support Context-Aware Ubiquitous Agent System

He Qiu-sheng and Tu Shi-liang

Department of Computer Science and Engineering, Fudan University  
Shanghai 200433, China  
{heqiusheng, sltu}@fudan.edu.cn

**Abstract.** Autonomous agents or semi-autonomous agents feature largely in their dynamic adaptation of its behaviors for changing environments to achieve some set of goals, especially in ubiquitous environments. It demands mechanism for coherently satisfying agent goals depending on changing availability of resources one the fly. Leveraging context-aware techniques and agent-oriented approaches, the paper proposes a lightweight architecture to build adaptive but predictable goal-driven ubiquitous agent systems based on OSGi (Open Service Gateway initiative), an open standard service-oriented framework. The proposed platform not only supports context acquisition, discovery and reasoning, but also provides a centric goal resolution mechanism using goal-tree to automatically specify service components for the satisfaction of agent goals. A Context Pair Language and the production rule are used as semantic basis to model contextual information as well as the goals.

## 1 Introduction

Now we are dealing with ubiquitous computing which was predicted by Mark Weiser in 1991 [1]. Recently, some research has been done to develop intelligent agent systems for ubiquitous environments. Since a ubiquitous environment is less predictable, ubiquitous agents should be adaptive to changing environments and work together to achieve some set of goals which represent user requirements. Hence, a ubiquitous agent is inherently context-aware. However, Developing context-aware ubiquitous agent systems present challenges to the research community. Firstly, development environments of ubiquitous agents are very different with their runtime environments. The former is typical a personal computer, while the latter is various resource-constrained devices. It is hard for the developer, in the development time, to anticipate precisely the available contextual information and appropriate behaviors of agents under the operation environments. As a result, a more promising approach is to have the agent express user requirements as goals. And then the multi-agent system automatically satisfies those goals, on the fly, by assembling software components, which utilizes the resources currently available to the user. Furthermore, an agent on ubiquitous devices has to be very small due to the nature of those devices. Even if an agent is small, it should be intelligent enough to satisfy its goals in a timely fashion. It can benefit from a dedicated architectural framework providing a reasonably mechanism for goal resolution, which probably get a better tradeoff between the autonomy

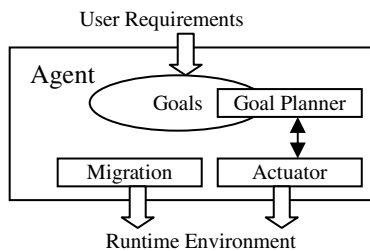
of a ubiquitous agent and its simplicity. In addition, a ubiquitous agent has to be platform independent due to the large variety of runtime environments, so that a middle-ware-built-in architecture as well as Java model is desired.

Following the above architectural principles, this paper proposes *lcUAS*, the lightweight context-aware ubiquitous agent architecture, to offer a general-purpose platform for engineering ubiquitous agent systems. The core of *lcUAS* is an architectural framework based on OSGi, a lightweight SOA (Service-Oriented Architecture) platform for deployment of Java-based services over wide area networks to local networks and embedded devices [2].

The rest of this paper is organized as follows. The paper first describes the software architecture of our ubiquitous agent. Next, it presents design and architecture of *lcUAS*, including the mechanism of modeling context and production rule-based goal as well as goal reasoning techniques. Section 4 discusses the demonstration prototype. Section 5 presents an overview of related work. Finally, Section 6 concludes the paper with future work.

## 2 Ubiquitous Agent

In this architecture, a ubiquitous agent is a persistent entity implemented by components written in Java programming language. It is a service provider as well as a requestor that plays one or more particular roles in a society of agents. Since agents are components, communication and cooperation between agents is purely syntax-based, i.e. using methods provided by Java classes. Roughly, ubiquitous agents involving in this architecture could be catalogued as *primitive agent*, *system agent* and *dedicated agent*. Primitive agents such as context sensing agent are the closest entities to the runtime environment. System agents provide runtime system services and they are the major part of *lcUAS*.



**Fig. 1.** Dedicated Ubiquitous Agent Architecture

By contrary, the dedicated agent is dynamically deployed on the environments. It consists of a goal list, goal planner, actuator and facility for agent migration (see figure 1). In general, high-level user requirements are described by Prolog terms called goals. A goal can be achieved, during runtime, by one of alternative plans that are composed by sub-goals. A sub-goal is series of production rules that represents agent actions out of abstract services provided by other agents (seen section 3). It is

goal planner’s responsibility to decompose goals into sub-goals. Nevertheless, because of limited resources of ubiquitous devices, in general, an agent’s goal planner generates plans during the development time or under a particular administrative environment before it is deployed into those devices. Using the services provided by system agents, the actuator added those sub-goals into a centric goal base. Next, run-time system dynamically resolves the goals and binds it to proper agents depending on current context. If the actuator knows that the platform can not find appropriate agents to resolve its goal, it is the goal planner’s duty to select alternative plan. Besides, the dedicated agent can provide services for other agents. Since a ubiquitous agent is written in Java language, it could be easily fit into the OSGi framework as a bundle, a standard Java JAR file using by OSGi to deploy services. OSGi also provides sophisticated mechanism for management of bundle lifecycle, so a dedicated agent easily migrates to any lcUAS-based system.

### 3 System Architecture

lcUAS is a set of tools and services aiming to provide a highly modular and flexible research grade context-aware multi-agent platform. Briefly, there are three key elements: a context modeling layer, a goal reasoning layer and OSGi-Compliant framework (see figure 2).

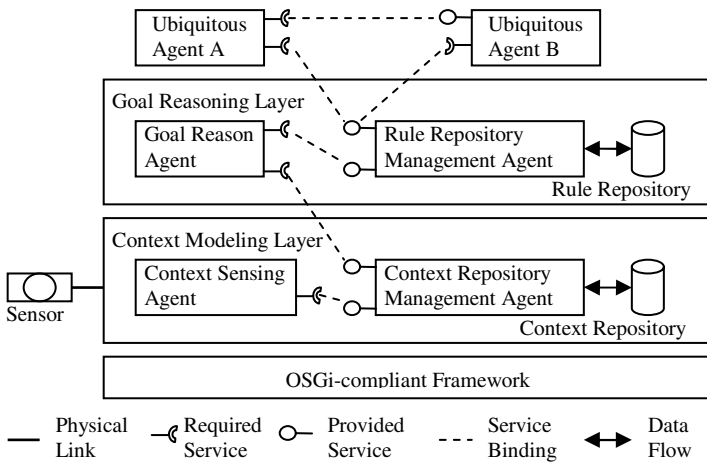


Fig. 2. The lcUAS Architecture Overview

Conceptually, context can be catalogued in two lays: low-level context such as location and high-level context such as user activity. lcUAS uses attribute-value pair-based context modeling language CPL (Context Pair Language) which grammar is rather similar to LDAP (Lightweight Directory Access Protocol) [3]. The major reason for the model is that OSGi represents the service properties and parameters of service LDAP filter as attribute-value pair too, so that it easily creates context-aware

applications on the top of the OSGi platform. Moreover, CPL represents context in a human-readable form which ease programming. Finally, resource constrained devices require relevant simple context model such as CPL.

A CPL expression, using a prefix format, presents one of well-defined attributes in a specific domain along with a range of available values for the attribute and an optional timestamp. For example, “((= *user\_location* inside) 10)” means user stay in house. A centric context repository stores all of contextual information that is acquired by sensing agents that drive external devices such as sensors. It is managed by a public agent named ContextMgr that provides a set of APIs for other agents to access context knowledge. Likewise, it generates high-level context from the low-level raw context, as well as resolves context conflicts and maintains knowledge consistency.

CPL is also used to represent goals, which represent policies of the system and describe tasks to be performed at runtime according to the available services in changing environment. A sub-goal is achieved by the fire of resolving production rules that are simple and represented in Java-based syntax style. It consists of an antecedent part (i.e. condition) and a consequent part (i.e. action). The former is a CPL expression, and the latter is set of action string, typically references to specific service objects, or program code that could occur in a regular Java method operating on particular objects. The following example shows a simple goal: three services, such as siren, camera and video player will be activated immediately by a contextual event that a smoke detector is triggered. In this example, the second part of condition “(OR (= *user\_location* inside) (AND ((NOT (= *user\_location* inside)) (= PDA available))))” implies that we should choose the composite services depended on current user location. For instance, if the user stays in the car, specific local device such as a vehicular gateway’s speaker will sound the alert and the video will be forwarded to a player connected to the vehicular gateway. If the user left the car, however, a certain hand-held device such as a PDA will provide those services, for example, that it will be caused to vibrate.

#### Example of a production rule

```
import lcUAS.sample.Siren;
import lcUAS.sample.Camera;
import lcUAS.sample.VideoPlayer;
Goal: AlarmBySmokeDetector
  Objects:
    Siren      s;
    Camera     c;
    VideoPlayer v;
  IF:
    (AND((= smoke_detector_state triggered)0)
      (OR(= user_location inside) (AND((NOT
        (= user_location inside)) (= PDA available)))));
  THEN:
    s.alert();
    c(v);
End
```

The heart of goal layer is a centric repository storing sub-goals. The goal repository management agent RuleMgr provides APIs and command tools for other agents to operate on goal knowledge. It is also RuleMgr's duty to load and compile specific production rule script on the fly.

An agent GoalReason is developed to reason over goals. Satisfaction of a top-level goal involves its representing rules, available services, and a process of evaluation and the heuristic selection of target services. Due to the nature of ubiquitous environments, it is most likely to be several goal rules fired at a given moment which could lead to potential conflicts. We provide several different conflict resolution policies such as first-order, LRU (Least Recently Used) and priority. User is allowed to write his own policy too. In practice, the agent GoalReason uses a goal-tree representing the context (internal nodes) and rules (leaf nodes). A real-time heuristic algorithm [4] using first-order logic and forward chaining selects a path that represents the most suitable action and providers of composed services according to currently given context.

## 4 Demonstration Prototype

We have built lcUAS on top of the OSGi service platform. Agent ContextMgr, GoalReason and RuleMgr as well as dedicated agents and context sensing agents are packaged as OSGi bundles and published specific OSGi services.

A prototype for telematics services is constructing based on our proposed architecture. One of its demonstration applications is a security system with additional navigation service. Currently, several implementations of OSGi are available. Of these, we use an open source implementation Oscar [5] because of its small footprint. We implement those system agents mentioned in section 3 as well as several context sensing agents that provide user information and car location and state of the security detectors respectively.

## 5 Related Work

O2S let user express their needs as goals which will be satisfied by assembling an implementation out of a set of generic components called pebbles during the runtime [6]. It seems to lack intelligence and a sophisticated mechanism to manage components. Kenta Cho and his colleagues develop an intelligent mobile agent named picoPlangent for use with portable devices [7]. The picoPlangent uses a built-in goal tree to plan and resolve goals, so that the agent developer should be aware of the details of runtime environments. Gulliver's Genie is a context-aware tourist guide using PDA that assists roaming tourists based Agent Factory platform (AF) [8]. The research try to provide a particular out-door context sensitive tourist guide. Apricot agent platform aims to develop context-aware and personalized mobile services [9]. However, it has heavy-weight agents, so that it might be unsuitable for some ubiquitous devices with limited resources. Several projects focusing on the ubiquitous agent area such as EasyMeeting room [10] and ACAI [11] have been developed to actively and adaptively serve users by leveraging ontology semantics. This approach is



flexible and expressive. However, even in a pretty simple ubiquitous computing environment, there is variety of information so that it needs skilled users to design ontology. In addition, ontology-based context modeling and reasoning tasks often ought to run on resource-rich devices.

## 6 Conclusion

Leveraging the techniques of agent technology and context-aware computing paradigm, lcUAS offers an architectural framework for engineering context-aware ubiquitous agent systems. With lcUAS, we are able to dynamically but predictably assemble an agent-based application out of service components as needs and context dictate. Being context-aware and performing agent-driven, we believe it will reduce the complexity of programming and system administration. In the other hand, the lcUAS also help to address some of the unique challenges created by the service-oriented approach for application developers, for example, service dependency management and service composition of dynamically available services.

Currently, lcUAS is only used in demonstration prototype. It still needs to prove its usefulness by being applied in real world situations. In addition, we need look into more sophisticated mechanism to reason goals.

## References

1. M. Weiser. The Computer for the 21st Century. *Scientific American*, Vol. 265, Issue 3: 94–104, September 1991.
2. The OSGi Alliance. OSGi Service Platform Release 4, 2005.
3. T. Howes. The String Representation of LDAP Search Filters. IETF, RFC 2254, Dec. 1997.
4. R. Korf. Real-time Heuristic Search. *Artificial Intelligence*, Vol. 42: 197–221, 1990.
5. The Oscar Homepage. <http://oscar.objectweb.org/>, 2005.
6. U. Saif, H. Pham, J. M. Paluska, et al. A Case for Goal-Oriented Programming Semantics. In: *System Support for Ubiquitous Computing Workshop at Ubicomp*, Seattle, WA, 2003.
7. K. Cho, H. Hayashi, M. Hattori, et al. PicoPlangent: An Intelligent Mobile Agent System for Ubiquitous Computing. In: *7th Pacific Rim International Workshop on Multi-Agents*, LNAI 3371: 43-65, Springer-Verlag Berlin Heidelberg 2005.
8. G. M. P. O'Hare, M. J. O'Grady. Gulliver's Genie: A Multi-Agent System for Ubiquitous and Intelligent Content Delivery. *Computer Communications*, Vol. 26, Issue 11: 1177-1187, Elsevier Press 2003.
9. P. Alahuhta, H. Löthman, H. Helaakoski, et al. Apricot Agent Platform for User-Friendly Mobile Service Development. In: *Proceedings of the 18th Int'l Conf on Architecture of Computing Systems*, LNCS 3432: 65-78, Springer-Verlag Berlin Heidelberg 2005.
10. H. Chen, T. Finin, A. Joshi, et. al. Intelligent Agents Meet the Semantic Web in Smart Spaces. *IEEE Internet Computing*, Vol. 8, Issue 6: 69-79, 2004.
11. M. Khedr, A. Karmouch. ACAI: Agent-Based Context-Aware Infrastructure for Spontaneous Applications. *Journal of Network and Computer Applications*, Vol. 28: 19-44, Elsevier Press 2005.

# Towards an Agent-Based Robust Collaborative Virtual Environment for E-Learning in the Service Grid

Changqin Huang<sup>1,3</sup>, Fuyin Xu<sup>1</sup>, Xianghua Xu<sup>2</sup>, and Xiaolin Zheng<sup>3</sup>

<sup>1</sup> College of Educational Information and Technology, South China Normal University, Guangzhou, P.R. China

<sup>2</sup> College of Computer Science, Hangzhou Dianzi University, Hangzhou, P.R. China

<sup>3</sup> College of Computer Science, Zhejiang University, Hangzhou, P.R. China  
cqhuang@zju.edu.cn, xufy@scnu.edu.cn,  
xianghuaxu@yahoo.com.cn, xlzheng@zju.edu.cn

**Abstract.** This paper proposes an Agent-based Collaborative Virtual Environment (ACVE) architecture using grid technologies. In this virtual environment, all e-learning resources and web services are bound via service encapsulation, and a special collaboration service layer undertakes the robust collaborative learning, moreover, the agents and mobile agents are applied to autonomous collaboration, and to management persistency of the virtual world. As for collaboration, the Locking Service, Context Awareness Service, etc are introduced to maintain efficiency of collaboration activities; two agents of learning roles are responsible for actual learning interactions, and the Session Agent has the ability of migrating among hosts to finish learning goals as well as maximize resource utilization. The GT4.0, JADE and a WS proxy are used to implement all functions. The result suggests it will be a more scalable and robust collaborative learning architecture.

## 1 Introduction

E-learning is currently taking on new characteristics like 'a collaborative group activity' [1]. The advent of grid technologies [2] brings the novel opportunity for students and teachers/administrators to communicate and interact with a variety of learning resources. The Grid offers unified resource-sharing environments and virtual communities by coordinating the resources and user/service activities. However, how to efficiently collaborate learning activities is a fundamental and hard-solved issue. While agent systems provide the Grid more flexible and agile to complement its shortage [3]. In this paper, we present an Agent based Collaborative Virtual Environment (ACVE), where the collaboration mechanism is specially constructed according to WSRF standards, and the agents are applied to improve intelligent interaction of learning activities.

The paper is organized as follows. We first review related work. In next section we describe an ACVE oriented grid architecture. Following this, we describe the enacted collaboration virtual environment in Section 4, and present how this collaboration system implements with necessary automation and robustness in Section 5. Finally, we conclude this paper.

## 2 Related Work

C. Bouras et al.[4] present the design principles for virtual spaces and two different tools as solutions for supporting e-collaboration in web-based learning communities. However, the reliable virtual space sharing needs be fairly improved. CoAKTing [5] provides a motivating example of the incorporation of collaborating technologies on top of a grid structure; the function's autonomy and intelligence are of more limitations. The APPLE [6] proposes the use of the grid for group-centric and P2P for individual-centric information retrieval in e-learning. A major limitation is platform-dependent and less collaboration concerns. A Semantic grid-based E-Learning Framework (SELF) [7] defines e-learning specific application layers on the top of semantic grid-based support layers to meet education applications, whilst, collaborative activities are well concerned with by software agents etc. But the heavyweight semantics discount the learning performance. I. Jorstad et al. [8] propose an excellent collaboration environment framework based on SOA, but the function flexibility and automaticity should be further considered. As I. Foster et al. [3] say, the grid and the agent need each other due to respective functional features. A. Negri et al. [9] present a flexible agent-based Grid system, and the agent-based collaboration is not proposed, while it is a part of our work.

## 3 An ACVE Oriented Grid Architecture

The e-learning system adopts a Service Oriented Architecture (SOA) based on the grid infrastructure; and a special collaboration layer is constructed, where agents are applied for autonomous, flexible collaboration behaviors. So the ACVE oriented grid architecture is proposed in Figure 1.

**Applications Layer.** The top layer - applications layer is oriented to e-learning users, and it implements various end-user applications: e-classroom, online discussion or learning/test, and related secondary work such as learning object/user management, learning object search, etc. this layer is supported by e-learning domain services suit and the service grid infrastructure.

**Domain Middleware Layer.** Common Services, Collaboration/Personalization mechanisms and Intelligent Agents are set according to the layered order. The first includes many public services in e-learning, such as service tracking, end-user/activity logging, application service combination/delivery, teaching and studying evaluation/analysis. Collaboration/Personalization mechanisms present some special services for collaborative and personalized learning activities respectively, and Intelligent Agents serve for these activities in order to improve the process autonomy and flexibility, moreover, both learning collaboration and personalization will apply these agents, and a few of agents are shared by them. Although Personalization Services may impart an important role by personalizing the individual centric information, they are not detailed for this paper does not concern. The collaboration services and related intelligent agents form the focused collaboration mechanisms for e-learning grids, which will be described in the following sections.

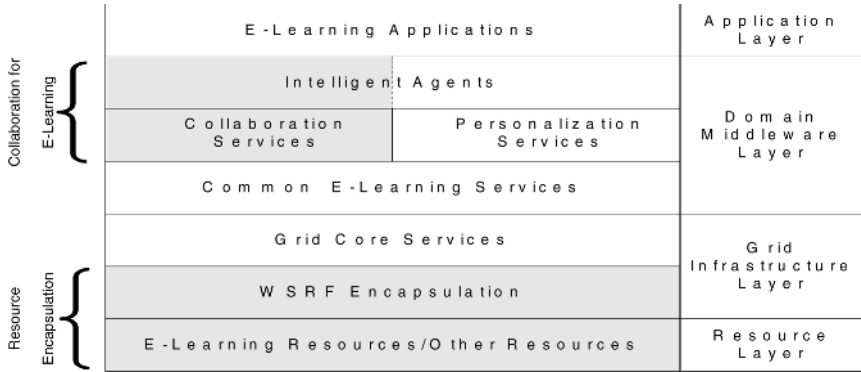


Fig. 1. The ACVE oriented grid architecture

**Grid Infrastructure and Resource Layers.** The Grid Infrastructure adopts SOA, and provides all core grid services (MDS, Delegation, ReliableFileTransfer, Security, Notification, Addressing, etc) and further advanced services (Resource Management, Job scheduling, etc). Resources are e-learning's bases, and whose formation plays an important role in increasing the resource availability. We use the new service grid criterions to do with these resources; thereby a special e-learning resource mechanism exists in the ACVE grid architecture. That is, the WSRF encapsulation module in the ACVE grid encapsulates the actual resources into E-learning Grid Resources (EGRs) in line with the WSRF standards.

## 4 An Agent Based Collaborative Virtual Environment

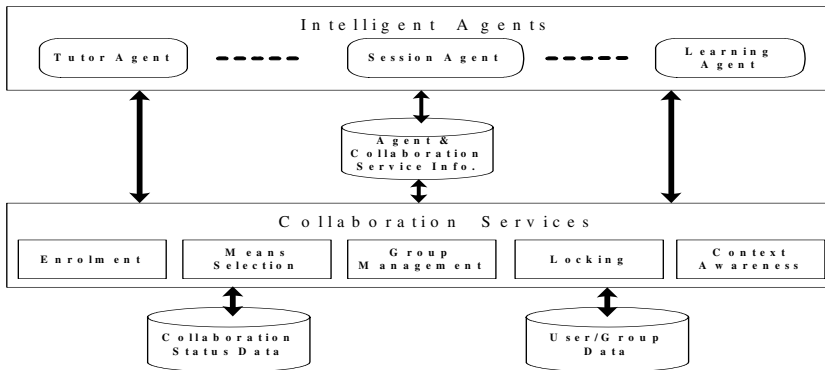
We apply grid services for constructing the fundamental collaborative elements, that will give e-learning better inter-operational and robust. Then, with the aid of agents and related components, the appropriate CVE model is depicted in Figure 2: As known from Figure 2, the collaboration model consists of collaboration work data, collaboration service, agent-service exchange information and three types of agents.

### 4.1 Collaboration Work Data

During collaboration, some work data should be stored. The collaboration status data repository is used to store the collaboration data in the execution of Collaboration Services, and these data have for example variable values, temporal parameters, register values, stack values, counting parameters, etc. User/Group Data refers to data, including the collaborative user roles, collaborative groups and the relations between the roles and the groups.

### 4.2 Collaboration Services

**Locking Service.** This service is necessary mechanisms to prevent the corruption of information resources. In general, there are several types of locks that can be



**Fig. 2.** The agent based collaborative virtual environment model

chosen for different situations: The Intent Lock meets the future intention of a user to acquire locks on a specific resource; The Shared Lock allows many users to access the same resources at the same time, however, no user is allowed to modify it; The Update Lock is acquired just prior to modifying the resource. **Enrollment Service.** In the CVE, the Enrollment Service is responsible for registering/subscribing a collaborating activity, giving a right decision when to participate in the activity. Of course, the relevant deregistration/un-subscription functions are granted while the participants/collaboration need(s). **Group Management.** The Group Management is responsible for defining different collaborative activities, and elaborating a setup for a collaborative activity to decide different applications and contents. At the same time, via a user call, this management service is in charge of the collaborative activity management. **Means Selection.** To give an appropriate way of communication to a certain collaboration activity, the Means Selection is enforced in this CVE as a grid service. For example, the asynchronous means (email, etc), the synchronous means (chat, online whiteboard, etc). **Context Awareness.** During collaborations, the state data varies while the interactions proceed, and the grid resources are autonomous and dynamic. Moreover, the changes are very vital to the current collaborations for activity adjustment; thereby, it is necessary to sense the context and give the appropriate notification.

### 4.3 Agent and Collaboration Service Information

To cooperate each other, agents and collaboration services need exchange work information. Like the middle-media, Agent & Collaboration Service Information should be equipped. It contains collaboration services advertisement and service instance information across the network; active agents and related agent information, invocation relations, etc; of course, the collaboration topic meta-data and global learning policies are yet necessary stored objects.

### 4.4 Intelligent Agents

**Tutor Agent.** The original concept of tutor-agent is the role of tutor during the learning. The 'tutor' is a functional concept, so it may be mapped to a teacher or a

student as a question solver in practice. This agent will autonomously do the following work: helps the learning agent to reach some cognitive goal, whose action methods consist of giving direct answers, posing heuristic questions, and advising a explorative learning contents, etc; analyzes the students' answers; records the answers from all the students and the communications among students; manages all participants in this administrative domain. **Session Agent.** It is responsible for completing a certain collaboration target as a session mediator. These services and resources, which are used in this certain collaborative activity, are distributed across the network and called by the Session Agents, so this kind of agent is the mobile agent. It can initiate a new tutor agent for every new session, and takes the tasks from tutor agents and travels around over the network while these tutor functions are required. In case that it finds appropriate services or other tutor agents remotely, these carried tasks are executed, and the relevant results or resulted reference are returned to the original tutor agent; Of course, it has to communicate with a human-tutor and get from him/her proper answers when the appropriate services cannot be found or last results cannot be gained after all called tutor agents finish relevant tasks. **Learning Agent.** The students' learning activity is of interaction and exploration, and it will delegate the learning agent. In general, this agent will adaptively carry out the following jobs according to learning requirements: requests to join in a certain CVE; presents and answers questions with the helps of the original students; automatically searches learning resource or executes the learning process advices from the corresponding tutor agents; adjusts learning activities according to the feedback from tutor agents or himself; sends necessary medial information and last data to the relevant student.

## 5 Implementation

Globus Toolkit 4.0 (GT4.0) is currently de facto grid middleware based on WSRF, and by which this agent based CVE infrastructure is implemented. As for detailed collaboration mechanism, the user authorization management calls the Community Authorization Service, and Intelligent Agents discover and monitor the associated Collaboration Services with the aid of MDS4, collaboration data is stored into the LDAP database. Context Awareness uses the Trigger Service [GT 4's plug-in] to sense context and notify the agents. All agents are built based on JADE. Three types of agents are respectively set some behaviors in terms of methods by extending classes. All agents and services communicate each other via the message mechanisms.

However, Agents and Grid Services live in different address spaces, and this fact brings some difficulties in integrating the Agent and the Grid environment, particularly, some mobile agents' mobility brings inconvenience in interacting each other. This system proposes a set of standard WS interfaces that an external system can invoke to access typical agent system functions, whilst, defines an integration interface by which agents can invoke WS components. As for the mobility of certain agents, a proxy interaction model is proposed, which implements a request-response interaction with the function of locating the current mobile agent and redirection of service results. The proxy exists in the form of a static WS, and can give adaptively two kinds of result delivery methods.

## 6 Conclusion

By combining the agents and grid services, we propose an Agent-based Collaborative Virtual Environment (ACVE) architecture. The available usage of e-learning resources is improved via resource service encapsulation, and a special collaboration service layer is constructed to maintain robust collaboration activities, moreover, the agents and mobile agents are applied to autonomous collaboration, and to management persistency of the virtual worlds. Finally, an applicable implementation is presented based on GT4.0 and JADE, and the interoperation between agents and grid services is dealt well with the aid of a WS proxy model.

**Acknowledgement.** The authors wish to thank the Scientific Research Fund of Hunan Provincial Education Department (Grant No. 04A037) and Zhejiang Provincial Natural Science Foundation (Grant No. Y105223), by which the research project has been supported.

## References

1. A. Romiszowski, How's the E-learning Baby? Factors Leading to Success or Failure of an Educational Technology Innovation, *Educational Technology*, 44(1): 5-27, Jan.2004.
2. I. Foster, C. Kesselman, et al., The Anatomy of the Grid: Enabling Scalable Virtual Organizations, *Jour. of High Performance Computing Applications*, 15(3): 200-222, 2001.
3. I. Foster, N. R. Jennings, et al., Brain meets brawn: Why grid and agents need each other. *Proc. of 3rd International Joint Conference on Autonomous Agents and Multi-agent Systems*, 1: 8-15, 2004.
4. C. Bouras, E. Giannaka, et al., Designing Virtual Spaces to Support Learning Communities and e-Collaboration, *Proc. of the 5th IEEE International Conference on Advanced Learning Technologies*, Kaohsiung, Taiwan, Jul. 2005
5. S.B. Shum, CoAKTinG, Collaborative Advanced Knowledge Technologies in the Grid, *Proc. of 2nd Workshop on Advanced Collaborative Environments*, Scotland, Jul., 2002.
6. Hai Jin et al., APPLE: A Novel P2P based E-Learning Environment, *Proc. of 6th International Workshop on Distributed Computing*, Kolkata, India, Dec. 2004.
7. Z. Abbas, M. Umer, et al., A Semantic Grid-based E-Learning Framework (SELF), *Proc. of the 5th International Symposium on Cluster Computing and Grid*, Cardiff, UK, May 2005.
8. I. Jorstad, S. Dustdar, D. V. Thanh, A Service Oriented Architecture Framework for Collaborative Services, *Proc. of the 14th IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprise*, Sweden, Jun. 2005.
9. A. Negri, A. Poggi, et al., Dynamic Grid tasks composition and distribution through agents, *Concurrency and Computation: Practice and Experience*, Current May 2006.

# Design of Music Recommendation System Using Context Information

Jong-Hun Kim<sup>1</sup>, Chang-Woo Song<sup>1</sup>, Kee-Wook Lim<sup>2</sup>, and Jung-Hyun Lee<sup>1</sup>

<sup>1</sup> Department of Computer Science & Engineering Inha University

Yong\_hyun Dong, Namgu, Incheon, Korea

{jhkim, whrma}@hci.inha.ac.kr, jhlee@inha.ac.kr

<sup>2</sup> Department of Knowledge Industrial Engineering Sunmoon University

Asansi, Chungcheongnam-do, Korea

rim@sunmoon.ac.kr

**Abstract.** Music recommendation systems used at the present time apply certain queries using appropriate music information or user profiles in order to obtain the desired results. However, these systems are unable to satisfy user desires because these systems only reply to the results of user queries or consider static information, such as a user's sex and age. In order to solve these problems, this paper attempts to define context information to select music and design a music recommendation system that is suited to a user's interests and preferences using a filtering method. The recommendation system used in this study uses an Open Service Gateway Initiative (OSGi) framework to recognize context information. Not only does this framework promote a higher user satisfaction rate for music recommendations, service quality is also improved by applying service mobility and distributed processing.

## 1 Introduction

In recent years, a recommendation system has been actively studied that predicts and recommends only information requested by a user. Various filtering methods [1] [2] are used in this recommendation system in order to estimate a user's preferences. For instance, a recommendation system using filtering methods have been used in some search engines such as Yahoo, Amazon online bookstore, and CDNow Internet shopping mall and have been well received by the user.

In this recommendation system, the similarity between the content of an item and user information was measured to recommend information desired by the user, and a content-based filtering method that based the rank on this measurement was also used. However, the recommendation of multimedia data is still limited [3] and is not highly reliable due to filtering only being based on static information. In particular, with a music recommendation system in the present web service environment, it is difficult to exactly recommend music that is desired by users because real-time context information like weather significantly affects a user's music selection.

Thus, this paper attempted to design a music recommendation system (MRS) that is able to recommend music according to a user's interests and conditions by



applying context information to a filtering method. In order to apply context information to a recommendation system, data regarded as a factor in music selection was configured as context information and was based on ontology. In addition, data obtained using various sensors and an Radio Frequency Identification (RFID) Tag based on Open Service Gateway Initiative (OSGi) could be recognized as exact context information through an ontology database and inference engine. The recognized context information in this process could be used in a filtering process and also applied to provide a recommended list that was well received by the user. The system used in this study consisted of three large sections; a Context Manager section that recognizes context, Service Manager section that continuously supports service even though the user moves to another location or a user device changes, and Music Recommendation Manager that recommends a music list based on the context information obtained in the Context Manager using a filtering method. The Context Manager and Service Manager were designed in an OSGi Gateway of home networks and the Music Recommendation Manager was implemented in a Recommendation Server.

## 2 Music Recommendation Using Context Information

This system defines contexts to recommend music by considering surrounding contexts and user information and configures a music list using a statistical filtering method. The statistical filtering used in a recommend process first configures a list by searching the music that corresponds to user preferences and user information from a Music Content Information Database (MCIDB) and builds an initial profile using this list.

The initial profile can be updated using the music title selected by users and the context information recognized in an OSGi environment in a music service. This profile is statistically analyzed and recommends a music list that corresponds to the context information from the MCIDB when a user inquires about a music service.

### 2.1 Configuration and Definition of Context Information

The configuration of context information for the MRS consists of user information (sex, age), pulsation, weather, temperature, and location information.

User information, pulsation, weather, temperature, and location information can be predefined as ontology, and data can be input from sensors. However, although weather information is predefined as ontology, its data can be established as a database retrieved from the Internet. Temperature data can be transferred using an OSGi framework and communication from a sensor used in wireless communication. User information and location information can be traced using an RFID Tag which is attached to a user's watch, and pulsation data can be obtained from a pulse sensor which is attached to a user's watch through real-time Zigbee communication.

Table 1 presents the definition of context information as different spaces, such as class 4 for pulsation, class 4 for temperature, class 7 for weather, and class 6 for location information, in order to build an ontology model.

**Table 1.** Configuration and Definition of Context Information

Sex	Age		Pulsation		Weather	Temperature		Location
	num.	class	num.	class	class	F°	class	class
MA- LE	0~ 7	Infant	0~ 40	Danger	Clear	-4~ 30.2	Cold	Balcony Bathroom Bedroom Guestroom Kitchen Livingroom
	8~ 11	Child	41~ 65	Low	Sunny Cloudy	32~ 68	Cool	
	12~ 17	Young Adult	66~ 120	Normal	Shower	69.8~ 86	Warm	
FEM- ALE	18~ 61	Adult	121~ 180	High	Rain	87.8~	Hot	
	62~	Old Adult	181~	Danger	Snow			
					Storm			

In particular, the service area is limited to homes, and the users' location is limited to the Balcony, Bathroom, Guestroom, Kitchen, and Living-room.

The context of the MRS based on the context information used in this study is defined as Web Ontology Language (OWL) that is used on a Semantic Web in order to configure and express exact contexts and various relationships.

**2.2 Filtering Method for Music Recommendation**

The system used in this study established the MCIDB with item contents and recommended music using the profile that corresponded with users.

To establish the MCIDB, an automatic establishing method in web documents and user input methods were used. In an automatic establishing method, web documents can be extracted by a web robot agent. In addition, the MCIDB can be built using the analysis of morphology. Table 2 presents an example of the MCIDB.

The earlier profile was configured as a title selected by the user according to the weather, temperature, and recommended ages from the MCIDB, and an additional inquiry profile was configured using a selected music list for a user's inquiry. In the case of the inquiry profile, the context information of weather, temperature, age, and pulsation becomes a music list selected by users.

In a recommendation method, the frequency of words is calculated using the morphological analysis of profile. Then, the word that presents the first highest frequency and music that agrees with the given keyword in the MCIDB are searched and configured as a list. This process is performed according to the profile that corresponds to context information when a user requires a recommended service. Also, the list ranks with the highest priority music that has the highest frequency in an inquiry profile and music that agrees with the given keyword in the MCIDB. In addition, the list can be configured in the order of weather, temperature, pulsation, and age profiles. All the duplicated music in the music list is deleted except for the music that has the highest ranking. Moreover, music selected by users can be recorded in an inquiry profile and that be used as an accurate music list according to the repetition in selections.

**Table 2.** Example of the Music Context Informaion Database

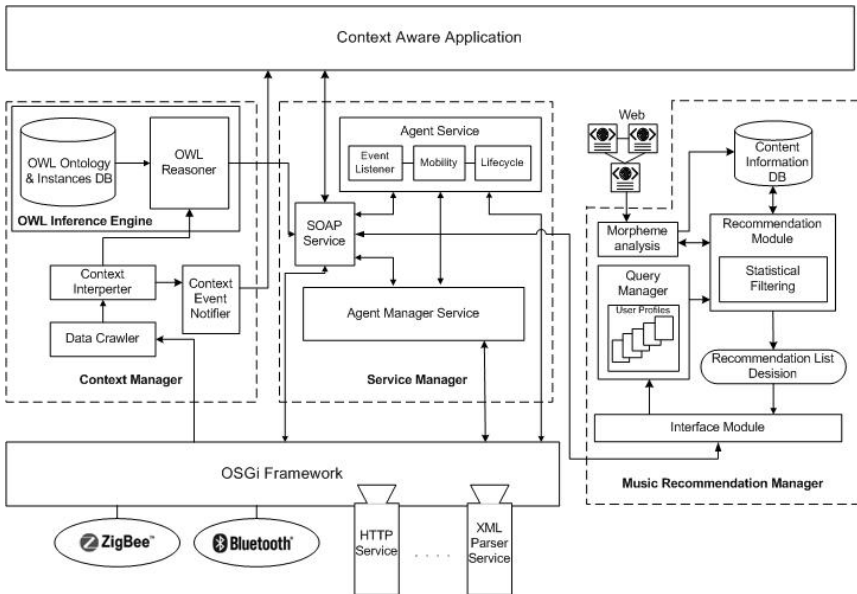
Music Information	Extracted Nouns
Title	Angel In The Snow
Singer	A-Ha
Genre	Pop
Age	Young Adult
Weather	Snow
Temperature	Cold
Keyword	Angel, Snow

### 3 System Design and Implementation

This chapter designed and implemented the MRS that was able to recommend proper music by estimating context information in a Java-based OSGi framework using the context definition and filtering method proposed in Chapter 2.

The OSGi is a type of industrial standard proposed by an OSGi organization in order to establish a standard connection method for Internet devices, such as household information devices and security systems [4].

Fig. 1 presents the diagram of the overall system. The MRS designed in this paper analyzed and suggested various data transferred from context recognition sensors and established it as information to recommend proper music through a filtering process for user profiles and the MCIDB. In order to perform this process, the music recommendation system consisted of a Context Manager, Service Manager, and Music Recommendation Manager.



**Fig. 1.** The Music Recommendation System Using Context Information

The Context Manager transferred data generated by events to a context analyzer and that data was transferred to an OWL inference engine. The OWL inference engine transferred data received from the context manager to the Service Manager in which data was transformed as information using an OWL inferencer including OWL ontology object database. The Service Manager consisted of a Bundle Service that provided music recommendation service as a bundle in a Simple Object Access Protocol (SOAP) Service, OSGi framework installed device in order to transfer information received from the OWL inference engine to the MRS, and an Application and Bundle Manager Service that supported the management of the mobility of bundles. The Music Recommendation Manager played a role in the decision of an optimal music recommendation list in a recommendation module by applying the recommendation list that corresponded to certain context information received from the Service Manager with the user profiles and the MCIDB to a filtering process in a recommendation module.

The system proposed in this study used an ontology inferencer Jena 2.0 and developed an OSGi gateway using the Knopflerfish 1.3.3, an open architecture source project which implemented a service framework.

## 4 System Evaluation

In the paper, survey is used to evaluation of system. 50 users participated in system estimation. Users responded by value to 1 - 5 on question of table 3. The test environment consisted of the Context Manager and Service Manager as a bundle in an OSGi gateway on a home network in which the Music Recommendation Manager was implemented on a desktop PC. In addition, the MCIDB consisted of 300 pop songs. Table 3 showed that user's satisfaction for system is high on the whole.

**Table 3.** Questionnaire and Results

Questionnaire	Result (sum of users' value/50*20)
Does the weather influence in music selection?	75.0 %
Is music that recommend satisfied?	90.8 %
Do you think that MRS is useful?	92.0 %

## 5 Conclusions and Future Work

The existing recommendation system only recommends data by analyzing it from the searching of simple inquiries or user preferences in a passive manner. This system cannot provide a proper search result that corresponds with user context and causes inconveniences to the user because the system requires too many opinions from the user in order to increase user satisfaction.

This paper actively obtained and recognized user context information and designed the Music Recommendation System (MRS) that is appropriate to the user using the recommendation and inquiry of the recommendation system. The context introduced in this study was defined as ontology in order to use it as context information. In

addition, a filtering method, which was based on a statistical method, was developed. Also, this system was designed based on an OSGi framework in order to provide services without any interruptions wherever and whenever in a home network and that increased user satisfaction.

As a future study, a new filtering method that considers user information to solve a specialization trend, which doesn't exceed the limited preferences of the MRS, is required.

## Acknowledgement

This research was supported by the Ministry of Information and Communication (MIC), Korea, under the Information Technology Research Center (ITRC) support program supervised by the Institute of Information Technology Assessment (IITA) (IITA-2005-C1090-0502-0031)

## References

1. M. Balabanovic, and Y. Shoham : Fab: Content-based, Collaborative Recommendation. *Communication of the Association of Computing Machinery*, Vol. 40, No. 3 (1997) 66-72
2. K. Y. Jung, and J. H. Lee : User Preference Mining through Hybrid Collaborative Filtering and Content-based Filtering in Recommendation System. *IEICE Transaction on Information and Systems*, Vol. E87-D, No.12 (2004) 2781-2790
3. H. -C. Chen, and A. L. P. Chen : A music recommendation system based on music data grouping and user interests. *Proc. of the CIKM'01* (2001) 231-238
4. P. Dobrev, D. Famolari, C. Kurzke, and B. A. Miller : Device and Service Discovery in Home Networks with OSGi. *IEEE Communications Magazine*, Vol. 40, Issue 8, August (2002) 86-92
5. P. J. Brown, J. D. Bovey, and X. Chen : Context-Aware Application: From the Laboratory to the Marketplace. *IEEE Personal Communication* (1997) 58-64
6. Wu, Y.H., Chen, Y.C., and Chen, A.L.P. : Enabling Personalized Recommendation on the Web Based on User Interests and Behaviors. In *Proceedings of IEEE International Workshop on Research Issues in Data Engineering (RIDE)*, (2001) 17-24
7. K. Romer, T. Schoch, F. Mattern, and T. Dubendorfer : Smart Identification Frameworks for Ubiquitous Computing Application. *IEEE International Conference on Pervasive Computing and Communication* (2003)
8. T. Gu, H. K. Pung, and D. Q. Zhang : An Ontology-based Context Model in Intelligent Environments. *Proceedings of Communication Networks and Distributed Systems Modeling and Simulation Conference* (2004) 270-275
9. S. Lee, S. Lee, K. Lim, and J. Lee : The Design of Webservices Framework Support Ontology Based Dynamic Service Composition. *Proceedings of the Second Asia Information Retrieval Symposium, LNCS 3689* (2005) 721-726

# Semantic Grid: Interoperability Between OWL and FIPA SL

Maruf Pasha<sup>1</sup>, H. Farooq Ahmad<sup>2</sup>, Arshad Ali<sup>1</sup>, and Hiroki Suguri<sup>2</sup>

<sup>1</sup> NUST Institute of Information Technology, 166-A, Street 9,  
Chaklala Scheme 3, Rawalpindi, Pakistan  
Phone: +92-51-9280658; Fax: +92-51-9280782  
{56maruf, arshad.ali}@niit.edu.pk

<sup>2</sup> Communication Technologies, 2-15-28 Omachi Aoba-ku,  
Sendai 980-0804 Japan  
Phone: +81-22-222-2591; Fax: +81-22-222-2545  
{suguri, farooq}@comtec.co.jp

**Abstract.** Evolution of World Wide Web has made it a focal point of research to apply structured and well-defined meanings to achieve Semantic Web and ultimately the Semantic Grid. The realization of this goal gives rise to the integration of technologies that can negotiate and cooperate in which agents have proven to be effective. OWL (Web Ontology Language) is currently the W3C standard for providing the explicit semantics to the web services whereas the FIPA Semantic Language is the core of agent platforms due to its high expressive power. The key objective of this paper is the development of architecture and semantic translations in such a way that the agents can communicate with the web services in an efficient manner.

## 1 Introduction

The evolution of semantic web will give rise to structure the meaningful contents of WebPages and it will be an extension of the existing web in which the information processed is given a well-defined meaning and better-enabled computers [1]. To efficiently discover and utilize the resources on the web, the semantics would act as epoxy resin for autonomy. To bracket together the explicit semantic representations in the web services several ontologies and ontological web languages have been anticipated, which not only have deficiencies but also impose an overhead. W3C and other research groups are doing a lot of efforts to standardize these languages in order to integrate the semantics in existing languages. In order to allow sharing and reuse of ontologies on the Semantic Web, a common ontology language is required. The W3C has proposed two ontology languages for use on the Semantic Web. The first is RDFS [6], based on XML [9] and logic programming, which is a lightweight ontology language. Whereas OWL [7] is another language by W3C with more expressive ontology language based on Description Logics [5][2].

The concept of the Semantic Web [1] has provided the foundations for the autonomous interoperation between the entities on the Internet. This interoperability can be achieved through annotation of the contents at arbitrary locations on the Web with machine processable data. When such annotations are associated with the ontologies, then computers can attain a certain degree of understanding of the data. Ontologies [4] are formal and explicit specifications of certain areas and are collective between large groups of stakeholders and are core of the Semantic Web. These properties make ontologies ideal for machine processing and facilitating interoperation and thus enhancing the interoperability and machine understandability.

On the other hand (MAS) Multi Agent System is a promising field of distributed artificial intelligence, where an agent is a computer system capable of flexible autonomous action in a dynamic unpredictable and open environment [8]. The exploitation of software agents in daily life application is increasing day by day and is becoming essential roles in the heterogeneous environments.

## 2 Abstract Architecture

Our devised model provides the semantic interoperability in distributed environments where technologies like agent applications and grid systems are combined and reused to achieve the autonomous semantic grid and thus providing a service oriented framework.

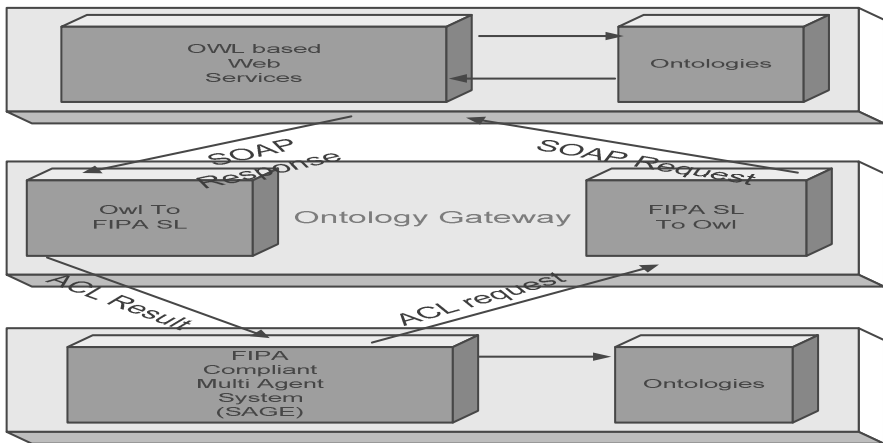


Fig. 1. Proposed System Architecture

As we are specifying that the OWL will be the W3C standard for specifying the services on the web and will act as a content language. The communication infrastructure specified by FIPA permit the agents to communicate using any mutually comprehensible content language as long as it fulfills a few minimal criteria as a FIPA compliant content language [FIPA, 2003].

In Figure 1 an abstract architecture of the proposed system is shown. The key idea is to show that how the software agents can communicate with the OWL based web

services in bringing the semantic operability, negotiation etc. The middleware act as a converter, which by taking input data (from the ACL message) converts it into mappings/translations described in OWL and vice versa.

### 3 Agents Communicating with the OWL Based Services

Fig 2. shows that whenever an Agent needs to search for a service on the GRID environment the transformation of the ACL based search query into SOAP based UDDI search query and forwards to UDDI where the required service is expected is

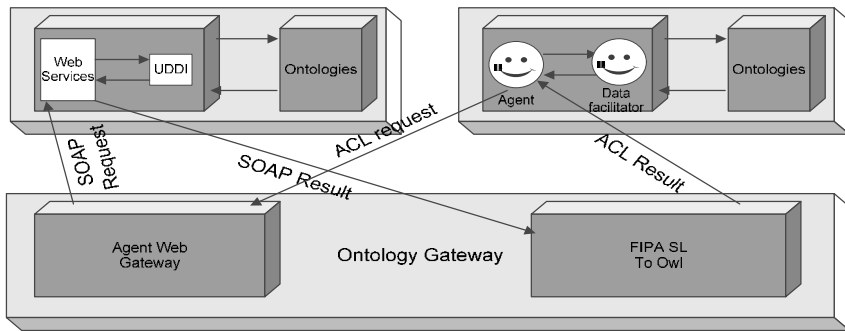


Fig. 2. Agents Communicating with Owl based services

done using the Agent Web Gateway’s component named ACL2SOAP converter [12] that is our earlier work. Now the Software Agent has come to know about the existence and the address of the required service. The address for the Web Service file is now retrieved and the Software Agent extracts the required Web Service. Software Agent is needed to know about the Ontology, AgentAction Schema, Predicate Schema and Concept Schema etc. so the Web Service is passed to Ontology Gateway that translates the Web service published in OWL into an equivalent FIPA complaint message.

### 4 OWL Based Services Interacting with the Agents

In this section, the shown detailed design enables W3C compliant SOAP based Whenever a Grid Entity needs to search for a service on the GRID environment the SOAP based UDDI query is passed to a protocol converter of Agent Web Gateway named SOAP2ACL. The protocol converter extracts out the service name sends a valid ACL based DF search request message to the Agent Platform. Directory Facilitator performs a search. If the required service is not found it is forwarded to a remote platform where the service is expected.

Directory Facilitator of the remote Agent Platform performs a search. If required service is found, the service is returned by the DF of that remote Agent Platform to



the agent at our middleware, which converts the FIPA Ontology into the OWL Ontology. And the service retrieved is embedded into the original SOAP message, which is forwarded to the Web Service Client that requested for the search. In this way, the Ontology Gateway helps the Web Service client to search for the services at Agent Platform.

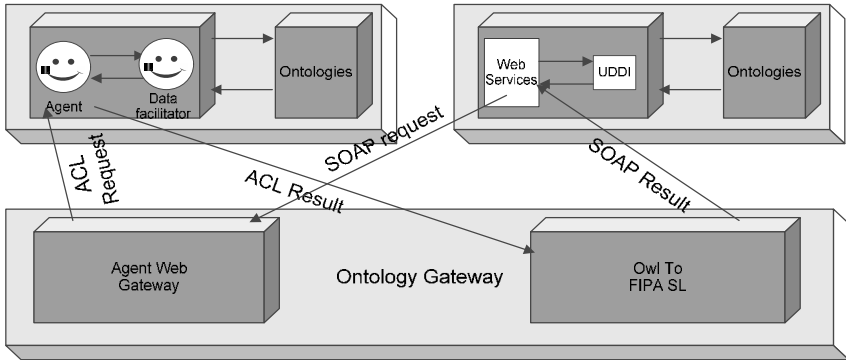


Fig. 3. Owl Based Services interacting with Agents

## 5 Analysis and Implementation

This section describes the analysis and implementation details that proposed system in which we have resolved the most important technical translations, keeping in view the specifications and standards of W3C OWL and FIPA SL, thus enabling communication and content transformation. According to FIPA ontology is composed of three types of schemas name concept, agent-action and predicate schema. Concept schema is used to define the user-defined objects in the agent’s ontology. Agent-Action schema contains the list of actions that the agent can be asked to perform whereas in Predicate schema in the Agents Ontology contains the list of possible outcomes or responses of the agents.

We have demonstrated this work through the implementation of a university ontology containing different classes, subclasses their properties and restriction on them and their details are given based on their relationships with each other.

In OWL class axioms are used to state that a class is exactly equivalent to, a subclass of, for the modality partial, the conjunction of a collection of superclasses etc its enhancement that is OWL DL is built with increased functionality of superclasses, some general restrictions and combination of Boolean expressions.

```
<rdfs:subClassOf>
    <owl:Class rdf:ID="courses" />
</rdfs:subClassOf>
```

The algorithm converts the above tag by extracting the class name i.e. courses and converting it to the SL equivalent that will be a concept

```
ConceptSchema coursesSchema =new ConceptSchema
("courses");
```

The resources in the class tag along with its attributes are converted into the concepts and the attributes are added as a part of schema.

```
ConceptSchema mastersSchema=new ConceptSchema("masters");
mastersSchema.add("totalStudents", stringSchema);
mastersSchema.addSuperSchema(coursesSchema);
```

The most challenging part was to deal with the predicates that are denoted in terms of object properties in OWL. These attributes were converted to there equivalent mappings in FIPA SL based on the restriction that they possess some values.

```
PredicateSchema hasStudentsSchema=new
PredicateSchema(hasStudents);
exPre.add(masters, hasStudentSchema);
exPre.add(student, hasStudentSchema);
```

In case of conversions from FIPA SL to OWL there were certain challenging obstacles, which were catered. In FIPA SL we can represent the different entities in terms of conceptschemas that is

```
ConceptSchema studentSchema= new
ConceptSchema("STUDENT");
ConceptSchema coursesSchema= new
ConceptSchema("COURSES");
```

Whereas in FIPA SL we can create the equivalent subclasses of the OWL by adding the superchema and specifying the base class name

```
ConceptSchema studentSchema = new
ConceptSchema("student");
studentSchema.addSuperSchema(UniversitySchema);
```

These FIPA SL base concepts and subconcepts were converted into equivalent OWL tags thus generating the code for the no of concepts with the superschema attached with them.

```
<owl:Class rdf:ID="student"/>
  <rdfs:subClassOf>
    <owl:Class rdf:ID="University"/>
  </rdfs:subClassOf>
</owl:Class>
```

Another big issue to deal with was the conversion of FIPA SL predicates into equivalent OWL mappings.

```
PredicateSchema studentOfSchema=new
PredicateSchema(STUDENT_OF);
studentOfSchema.add("courses", coursesSchema);
studentOfSchema.add("student", studentSchema);
```

Our implementation module generates their equivalent OWL mappings in terms of classes, restrictions and attributes.

```
<owl:Class rdf:resource="#courses" />
<rdfs:subClassOf>
<owl:Restriction>
<owl:onProperty>
<owl:ObjectProperty rdf:ID="studentOf" />
</owl:onProperty>
<owl:someValuesFrom>
<owl:Class rdf:resource="#student" />
</owl:someValuesFrom>
</owl:Restriction>
</rdfs:subClassOf>
```

We have successfully implemented these transformation/mappings without any information loss.

## 6 Conclusion

In this paper we have devised an architecture to provide a detailed insight of how the agents and web services can communicate with each other autonomously, initial translation/mappings for the interoperability of FIPA SL with the OWL in the context of achieving a semantic grid to achieve autonomous coordination in the messages that are exchanged between the FIPA compliant Multi agent systems and OWL based web services. We have done a detailed analysis of FIPA Semantic Language and the W3C OWL. Our future work will mainly focus on introducing new operators while transforming from SL to OWL as SL is semantically enriched.

## References

1. T. Berners-Lee, J. Hendler, and O. Lassila. The semantic web. *Scientific American*, 284(5): 34-43, May 2001.
2. Horrocks I. Reasoning with Expressive Description Logics: Theory and Practice. In: Andrei Voronkov, (ed) Proc. 18th Int. Conf. on Automated Deduction (CADE-18). Springer Verlag, pp.1-15, 2002.
3. J. Ferber. *Multi Agent Systems*, Addison Wesley, Reading MA, 1999.
4. D. Fensel. *Ontologies: Silver Bullet for Knowledge Management and Electronic Commerce*, 2nd edition. Springer-Verlag, Berlin, 2003.
5. F. Baader, D. Calvanese, D. L. McGuinness, D. Nardi, and P. F. Patel-Schneider, eds. *The Description Logic Handbook*. Cambridge University Press, 2003.
6. D. Brickley and R. V. Guha. RDF vocabulary description language 1.0: RDF schema. Recommendation 10 February 2004, W3C, 2004.
7. M. Dean and G. Schreiber, eds. *OWL Web Ontology Language Reference*. 2004. W3C Recommendation 10 February 2004.

8. M. Wooldrige, "Agent based Software Engineering IEE Proc. Software Engineering, vol. 144, no .1, pp.26-37, 1997".
9. H. Chen, Y. Chung, M. Ramsey, C.C. Yang, An intelligent personal spider (agent) for dynamic internet/intranet searching, *Decision Support Systems* 23 (1) (1998).
10. Zaheer Abbas Khan, H. Farooq Ahmad, Arshad Ali, Hiroki Suguri, China, April "Decentralized Architecture for Fault Tolerant Multi Agent System", ISADS, 04, 5-7, 2005 (2004) pp 167-174.
11. Abdul Ghafoor, Mujahid ur Rehman, Zaheer Abbas Khan, H. Farooq Ahmad, Arshad Ali, "SAGE: Next Generation Multi-Agent System", PDPTA'04, pp.139- 145, Vol. 1, (2004).s
12. Muhammad Omair Shafiq, Hafiz Farooq Ahmad, Hiroki Suguri, Arshad Ali, "Autonomous Semantic Grid: Principles of Autonomous Decentralized Systems for Grid Computing" IEICE/IEEE TRANS., VOL. E85-A/B/C/D, No. 1(2005).

# An Agent-Based Adaptive Task-Scheduling Model for Peer-to-Peer Computational Grids

Zhikun Zhao and Wei Li

School of Computer Science, Central Queensland University  
Rockhampton QLD 4702, Australia  
{z.zhao, w.li}@cqu.edu.au

**Abstract.** This paper presents an agent-based adaptive task-scheduling model for pure P2P computational grids, in which the task-scheduling mechanism is recursive, dependable and purely decentralized. The main idea is that the application provides the task decomposing method and decomposition is triggered by the platform once parallel running is possible. Mobile agents are used to carry tasks and results moving from one node to another. Each node has a manager agent administrating other agents to distribute and schedule tasks. From the preliminary testing results, it can be seen that the model can provide a low-cost large-scale computing platform on the pure P2P architecture. It avoids the disadvantages of hybrid-P2P architectures and is easy to adapt to the applications that can be decomposed into independent coarse-grained subtasks.

## 1 Introduction

Computational grid and Peer-to-Peer (P2P) computing are two important branches of distributed computing. Computational grid is an infrastructure that can integrate the computational resources of almost all kinds of computing devices to form a global problem-solving environment [6] P2P systems aim at resource sharing and collaboration through direct communication between computers without a centralized server as a medium [2].

Grid computing is first tested in the 1995 I-WAY experiment [7] and the current focus is around the Global Grid forum and the Globus project [5] P2P computing came into front as the appearance of SETI@Home and BOINC projects [3] Some people also call these kinds of systems public-resource computing [4] or volunteer computing [11] because they utilize the idle processor time on the computers that people volunteer to join the system.

The existing computational grids and P2P computing systems have their weaknesses. In a grid, users' clients are only used to submit tasks and display results and they will not contribute to the capability of the grid. While in P2P systems, users cannot submit applications to the systems directly because task distributing is under the control of the servers. In addition, both the existing grids and P2P systems need one or more centralized servers. It means that the costs of these systems are high and the halt of a server will cause the system or part of the system out of work.

We present a task-scheduling model in this paper. There need not any centralized server in the system. Each node can submit tasks and share its idle processor time with others. Each node can join or quit during runtime time, which will not cause failure of other nodes' tasks. A decentralized, recursive, and redundant task-scheduling mechanism is provided to support these features. Tasks are dynamically distributed to all the nodes adaptively. Agents are used to carry tasks and implement task scheduling as mobile agents can move from computer to computer.

Some researchers also introduce agent concept into P2P grids as agents and peers are naturally homogeneous [8][9][13][14] In these models, agents are the basic units to implement peer functions. And another related domain is Java-based adaptive parallel computing platform, such as P3 [10], Javelin [15] and HARNESS [12]. These platforms provide interfaces for applications to be implemented as tasks. The main difference between our model and these agent-based models and adaptive parallel computing platforms is that our model is built on pure P2P architecture while these models and platforms are built on hybrid- or non-P2P architectures.

From the preliminary testing results, we can see that our model has a steady load balance in face of the changes in P2P environment, such as node joining and quitting, node with heterogeneous and different computing capabilities, network topology changing, and other uncertainties. The model requires applications implementing the task-decomposing method and the result-composing method, and the subtasks independent and without large amounts of data.

## 2 The Task-Scheduling Model

### 2.1 Agent-Based Task Assigning

Mobile agent is an appropriate carrier for task to move from one machine to another. Mobile agent is in fact a program that can migrate from computer to computer in the network [16] The procedure of agent-based task scheduling is as follows: if node A needs to assign a task to node B, node A assembles a task agent with the task object. Then the task agent moves from node A to node B, and node B executes the task. Finally node B assembles a result agent with the result object after finishing the task, and the result agent returns from node B to node A. See Fig 1.

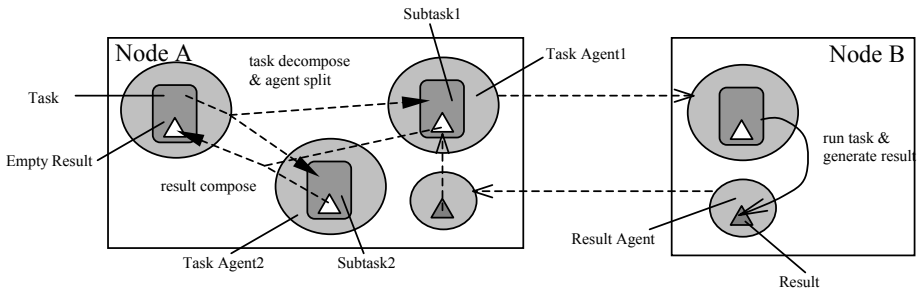


Fig. 1. Task decomposing, assigning, and result composing

A task should be decomposed into any amounts of pieces when needed, unless the computing granularity of the task is too small. So the task must implement the “decompose” method. Because our model is based on Java platform, “decompose” is as a method of interface “ParallelizableTask” that each task object must implement. On the other hand, the result is saved in another “TaskResult” object, which must implement the method “compose” to compose the results of the subtasks. These two interfaces form the main application interface.

The executing codes of a task are encapsulated in a Java class file, which is propagated among nodes through the codebase mechanism of Java virtual machine (JVM). The security is managed by the security manager of JVM. Tasks are limited to use the processor only, and have no right to access the local file system and the network except those of their initiators.

## 2.2 Task Distributing

Each node is managed by a manager agent, which administrates other agents. Task distributing is an adaptive process due to the dynamical P2P environment. Manager agent distributes tasks based on its own knowledge. The principle is to let the task run parallel at as more nodes as possible. So each manager agent follows the rule that parts of its local tasks should be distributed to its neighbors once its neighbors become free. If a node becomes free, the manager agent will inform all its neighbors. To avoid several neighbors assigning tasks to one node at the same time, a contract net protocol is used.

All manager agents collaborate together to distribute tasks in the system. For instance, there are four nodes in a grid, and the network topology is as Fig 2-a. Suppose they all be free when node A starts a task T. Node A looks up his neighbors and finds that node B and C are free. Then node A decomposes task T into three parts and assigns one part to node B and C separately, see Fig 2-b. Then the same process takes place in node B. B distributes half of his work to D, see Fig 2-c. Another condition is shown in Fig 2-d. Node E joins the grid when node A is executing the one-third part of the task. Node A distributes half of his task to his new neighbor. C seems undertaking more work, but C will continue delivering its work to others when they become free until the whole task is finished.

Because a node may quit the system during runtime, the tasks it undertakes maybe unable to finish. Therefore, nodes should continue to monitor the status of the tasks it

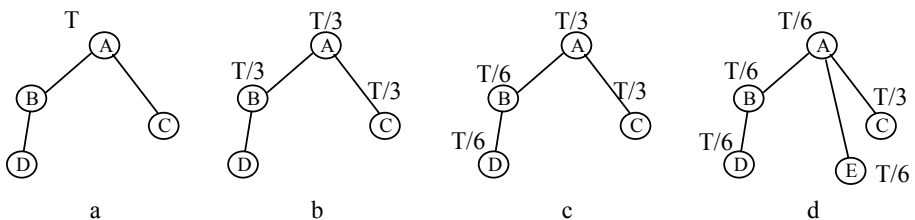
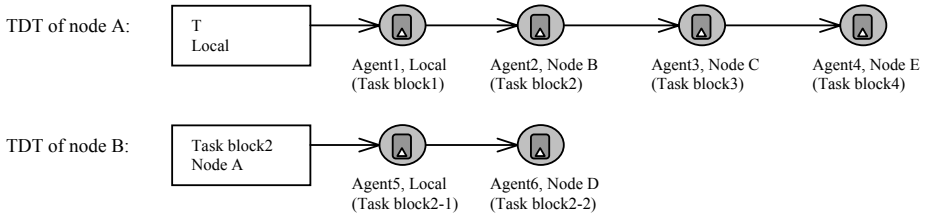


Fig. 2. Task distribution process

assigned out and should re-assign them when losing contact with the node that executes them. Thus a task agent distribution table (TDT) is necessary to maintain by the manager agent. TDT is a list that records all the task agents that take different parts of the task and where these agents are. For convenience, we name a subtask as a task block. For the example in Fig 2-d, the TDTs of node A and B are shown in Fig 3:



**Fig. 3.** The TDTs at node A and B for the example in Figure 3-d

The task executing state is recorded in the TDT, which is maintained by a heartbeat protocol. A task agent need reassign if its state is not updated for a period longer than a threshold. The manager agent may reassign it at the next scheduling process.

The results of task-blocks are also saved in TDT. A task is finished after all its subtasks are finished. Then all the results of the blocks are integrated together to generate the final result. And the final result is sent to the end user if the task is a local one or to the assigner otherwise. The result composing process is a reverse recursive process of task decomposing.

There is another condition need consider. A node may have completed its local task-block while other blocks are still unfinished. To utilize computing resource, the node should save the TDT of the current task and be available to start a new task. The unfinished tasks will queue and form an unfinished task list (UTL), in which the tasks may be finished some time in future or the unfinished blocks may be re-distributed.

A node is able to execute a new task when it becomes free, even though the new task maybe a part of his task blocks it assigned out before. The reason is that a node regards the task block assigned to it as an entirely new task. So the task distributing is an adaptive and recursive process, which can dynamically adjust the load balance among the nodes in the system.

### 2.3 Task-Scheduling Algorithm

There are two situations need task scheduling. One is that a user starts a new task. The other is that a node finishes its current block or finds a free neighbor.

In the first situation, the node can be in two possible states, busy or free. If it is busy, the new task is added to the UTL as an unfinished task. It will be a candidate when a free node appears. Here we use a first-come-first-served policy. If the node is free, the manager agent runs the new task at the local processor at once. Then it looks for free neighbors and decomposes the task into corresponding count of parts and assigns to them.



In the second situation, the manager agent need select a task agent from the UTL. The manager agent searches the UTL first for the oldest non-update task agent. If such an agent is found, it should be run locally or assigned to the free neighbor. Otherwise, we discuss it in two conditions. If it is the local processor becoming free, no task can be executed means the local node has nothing to do now. If it is a neighbor becoming free, the current task of the local node should be decomposed into two parts and one should be assigned to the neighbor.

The scheduling algorithm we provide is focus on the utilization of processors. Of course other methods can be adopt. First, tasks can be assigned priorities and tasks with high priority will be scheduled first. Second, task blocks may implement the interface of processing percentage and running time, then the shortest job first, shortest remaining time first, or other time related algorithms might be applied. Third, nodes can provide the parameter of computational capability, and then the task may be decomposed according to the capabilities of nodes rather than be separated averagely.

### 3 An Application Case Study: N-Queen Problem

To illustrate how to implement an application in our model, we use N-Queen problem as a case study. N-Queen problem is a classical space search problem [1]. N queens are placed on a chessboard with N rows and N columns so that there are not two queens at a line horizontally, vertically, or diagonally in the chessboard. Such a situation is called a solution. The total number of the solutions is the answer to the N-Queen problem.

In a solution, each queen must be placed at a different row So a state in the search space can be denoted as a N dimensional vector  $[q_1, q_2, \dots, q_N]$ , where  $q_i$  is the horizontal position of the queen at row  $i$  For example,  $[5,3,6,0,7,1,4,2]$  is a solution to the 8-Queen problem. The vector also can be regarded as a number based on unit of N. Thus the search space of 8-Queen problem is from  $S_0=[0,0,0,0,0,0,0,0]$  to  $S_8^8=[7,7,7,7,7,7,7,7]$ .

Suppose the search space of a task is  $S_{start}$  to  $S_{end}$  To decompose the task into M subtasks, each subtask should have a search space with length  $d=(S_{end}-S_{start})/M$ . The search spaces of the subtasks are  $(S_{start}, S_{start}+d)$ ,  $(S_{start}+d+1, S_{start}+d*2)$ , ..., and so on. Each subtask searches its space with a backtracking algorithm. The result is the amount of solutions in the search space. Result composition is calculating the sum of two numbers. In addition, the granularity of a task is a predefined minimum length of the search space.

The application works well in our model. Table 1 is some preliminary testing results under the condition of three computers with different hardware configuration. The computers are denoted as A, B and C. The data in the columns titled with A, B and C is the time cost when separately using one node to solve the problem. The data in the rest three columns are the results when using three nodes together but with different network topology. For example, "B-A-C" means B connects with A and A connects with C. The task is always started at node A. Under each topology, we do three tests. In the first test, we start all three nodes and then start the task. In second

test, we start the task at A and then start B after 10 seconds and start C after another 10 seconds. In third test, we begin with the same operations of the test 2, but we stop C after C starts 10 seconds.

**Table 1.** Some primary testing results (in milliseconds)

	A	B	C	B-A-C			A-B-C			A-B-C-A		
				Test 1	Test 2	Test 3	Test 1	Test 2	Test 3	Test 1	Test 2	Test 3
14-Queen	62188	82389	40031	21469	34682	36254	22478	34706	36109	21281	34937	36813
15-Queen	433921	574956	276453	138346	150524	259034	140414	158237	259549	135975	151021	260268
16-Queen	3612438	4270090	2025563	1007486	1020531	1973469	1028460	1033859	1974553	1009893	1023312	1975492

The theoretical value of the results of the first and second tests can be calculated using the following expressions:

$$T_1 = 1 / (1/A + 1/B + 1/C) \quad (1)$$

$$T_2 = (1 - 20000/A - 10000/B) / (1/A + 1/B + 1/C) + 20000 \quad (2)$$

We can see that the results in the column of ‘Test 1’ and ‘Test 2’ of all three conditions are close to the theoretical value. It preliminary proves that our model has a good load balance under the dynamical network circumstance. The result of the third test is hard to calculate in theory, but it illustrates that the model can work out the result even under the circumstance of losing some nodes. Large-scale tests and other applications will be carried out in the future.

## 4 Conclusion and Future Work

In this paper, we proposed an agent-based adaptive task-scheduling model for P2P computational grids. The model is built on pure P2P architecture. So a peer can work if only it connects to another peer in the system. Each node has the ability to start applications. And nodes are allowed to join and quit during runtime. And the model does not need the communicating protocol have broadcasting functions.

We have developed the prototype of the model and also some example applications in Java. It shows that our model works efficiently. It adapts to the applications that can be decomposed into independent coarse-grained subtasks. The future work includes: first, study other task-scheduling algorithms in the model, such as priority or time related. Second, explore mechanisms for executing application with large amounts of data because the model has a limitation that the size of the data that a subtask needs could not be too large.

## Acknowledgements

This work was supported by Central Queensland University, Australia under Research Advancement Awards Scheme (RAAS) grants, 2006~2007.

## References

- [1] Bah-Hwee Gwee, Meng-Hiot Lim. An evolution search algorithm for solving N-queen problems. *International Journal of Computer Applications in Technology (IJCAT)*, Vol. 24, No. 1, 2005
- [2] Barkai, D. *Peer-to-Peer Computing: Technologies for Sharing and Collaborating on the Net*. Intel Press 2002
- [3] David P.Anderson, BOINC: A System for Public-Resource Computing and Storage. In: *Proceedings of 5th IEEE/ACM International Workshop on Grid Computing*, Pittsburgh, USA, 2004
- [4] D.P.Anderson, J.Cobb, E.Korpela, M.Lebofsky, and D.Werthimer. SETI@home: an experiment in public resource computing. *Commun. ACM*, 45(11), 56--61, 2002
- [5] I.Foster and C.Kesselman, The Globus Project: A Status Report. In: *Proceedings of IPPS/SPDP 1998 (12th International Parallel Processing Symposium & 9th Symposium on Parallel and Distributed Processing)*, 4-18, Orlando, USA, 1998
- [6] I.Foster and C.Kesselman, editors. *The Grid: Blueprint for a New Computing Infrastructure*. Morgan Kaufmann, San Fransisco, CA, 1999
- [7] I.Foster, Internet Computing and the Emerging Grid. *Nature*, December 2000
- [8] Jin X., Liu J. The dynamics of peer-to-peer tasks: an agent based perspective. In: *Proceedings of Agents and Peer-to-peer Computing (2004)*, 84-95, New York City, USA, 2004
- [9] Li T.Y., Zhao Z.G., You S.Z. A-peer: An agent platform integrating peer-to-peer network. In: *Proceedings of IEEE International Symposium on Cluster Computing and the Grid (2003)*, 614-617, Tokyo, Japan, 2003
- [10] Licinio Oliveira, Luis Lopes, and Fernando Silva, P3: Parallel Peer to Peer An Internet Parallel Programming Environment Networking 2002 Workshops, LNCS 2376, 274-288, Pisa, Italy, 2002
- [11] Luis F. G. Sarmenta Volunteer Computing. Ph.D. Thesis, MIT, Department of Electrical Engineering and Computer Science, 2001
- [12] M.Migliardi and V.Sunderam. Heterogeneous distributed virtual machines in the HARNESS meta-computing framework. In: *Proceedings of Second Merged Symposium IPPS/SPDP 1999(13th International Parallel Processing Symposium & 10th Symposium on Parallel and Distributed Processing)*, San Juan, Puerto Rico, 1999
- [13] Moro G., Ouksel A.M., Sartori C. Agents and peer-to-peer computing: a promising combination of paradigms. In: *Proceedings of the 1st International Workshop on Agents and Peer-to-Peer Computing*. Bologna, Italy, 2002
- [14] Overeinder B.J., Posthumus E., Brazier F.M Integrating peer-to-peer networking and computing in the agentscape framework. In: *Proceedings of The 2nd IEEE International Conference on Peer-to-Peer Computing*, 96-103, Linkoping, Sweden, 2002
- [15] P.Cappelo, B.Christiansen, M.F.Ionescu, M.Neary, K.Schauser, D.Wu. Javelin: Internet-based Parallel Computing using Java. *ACM 1997 Workshop on Java for Science and Engineering Computation*, LasVegas, USA, June 1997
- [16] Pham V., Karmouch A., *Mobile Software Agents: An Overview IEEE Communications*, Vol. 36, No 7, 26-37, 1998

# Cluster-Based Secure Data Transmission Solution for Ad Hoc Network

Hwan-Seok Yang<sup>1</sup>, Joung-Min Kim<sup>1</sup>, and Seung-Kyu Park<sup>2</sup>

<sup>1</sup> Computer Science and Statistic Graduate School, Chosun Univ, Korea

<sup>2</sup> Dept. of Cyber Investigation Police, Howon Univ, Korea  
badhack@chosun.ac.kr, mrjjoung@hotmail.com,  
parksk@mail.howon.ac.kr

**Abstract.** This study proposes a cluster-based data transmission solution for ad hoc networks. In the proposed concept, when a cluster is formed, cluster head receives the trust values of neighboring nodes and evaluates the trust of cluster member nodes in an accurate and efficient way using trust values it received and its own leader value. This mechanism enables cluster head to authenticate new nodes randomly joining the cluster. In addition, a limited broadcasting mechanism was used by reducing the amount of control packets in an effort to enhance the efficiency in data transmission. The number of cluster heads, the communication path length and the amount of control packets were measured to verify the efficiency of the proposed concept.

## 1 Introduction

Ad hoc network has the advantage of making it easy to establish a network under the circumstance in which the establishment of a wired network is impossible. Ad hoc network is however more vulnerable to attacks given the nature of mobile network, lack of fixed infrastructure and dynamic topological changes [1]. The security solution for ad hoc networks needs to be designed in compliance with these characteristics. A safe authentication mechanism is the centerpiece of security system used for ad hoc networks[2][3]. It seemed like that typical security solutions are lackluster against new security threats. And new routing protocols based on the evaluation of the trust of each cluster member have been suggested[4][5][6]. In these models, each node evaluates the trust of its neighboring nodes on the basis of its experience with them. The security of the network is therefore closely linked to the trust management among nodes. Under this trust evaluation scheme, it is hardly possible for a node to evaluate the trust of nodes with which it has almost no communication. Moreover, a huge amount of resources is demanded to have a node store individual information of neighboring nodes.

In the proposed concept, a cluster is formed around a node and neighboring nodes in distance of 1-hop, and cluster head, selected by cluster member nodes, guarantees the trust of nodes and serves as certificate authority(CA). Each node transfers the trust values of its adjacent neighbors to cluster head. The trust of node is evaluated by information presented by cluster head and its own experience. When the node does not have its own experience, the trust of node is evaluated by information presented

by cluster head only. This means that node does not necessarily keep the information about other nodes within a cluster. In addition, the proposed solution aimed to enhance efficiency in data transmission in ad hoc networks by adopting a limited broadcasting scheme and reducing the amount of control packet.

## 2 Cluster-Based Data Transmission Mechanism

This study proposes a cluster-based mechanism designed for secure data transmission and high efficiency in ad hoc network. Authentication, which is the most critical part in the design of security solutions, is performed under the following conditions:

### 2.1 Cluster Formation

A cluster is created around a node and its neighboring nodes in distance of 1-hop. And node that has the highest trust value and number of link is selected as cluster head. The trust value indicates a success rate for delivery of packet by neighboring nodes and is measured on a scale of 0 to 10. The initial trust value is set at 1. If the cluster head turns out to be a member of other cluster, a node with the second highest trust value should be selected as cluster head. Once a cluster head is decided, gateway nodes and member nodes are also elected. And member nodes transfer the trust values of their neighboring nodes. If cluster head receives the trust value of the same node repeatedly, the sum of trust values is averaged and stored. Cluster head guarantees the trust of nodes within a cluster. That is, member nodes can request their own certificate from cluster head and communicate with other nodes with the certificate.

### 2.2 Authentication of Node Identities

Cluster head issues a certificate to all the nodes within a cluster. It issues a certificate to gateway node A based on the following equation:

$$\text{GwCert} := \{ \text{Node}(A), \text{trust value}, \text{PubKey}(\text{CH}), \text{Validity}(t), \\ \text{Status}(\text{"Gw"}), \text{Sign}(\text{CH'id}, \text{Create time}) \}$$

Where the trust value indicates the average of gateway node A's success rates for packet delivery, which were collected from cluster members. Validity(t) indicates expiry date, and gateway nodes request certificate renewal when validity date passed. Member nodes also receive the same type of certificate created in accordance with the following equation.

$$\text{MnCert} := \{ \text{Node}(B), \text{trust value}, \text{PubKey}(\text{CH}), \text{Validity}(t), \\ \text{Status}(\text{"Mn"}), \text{Sign}(\text{CH'id}, \text{Create time}) \}$$

The differences in certificates given to gateway nodes and member nodes lie in status values and the number of cluster heads which issue a certificate for them.

### 2.3 Mobility of Nodes

New nodes can join the cluster when node moves into another cluster leaving its own cluster and when a new node joins the network for the first time. On the first

occasion, the node transfer the certificate received in the former cluster-to-cluster head in the new cluster. Once new cluster head authenticate the node, it begins establishing the trust. On the second occasion, cluster head begins to collect information on the trust of the new node. The outgoing node sends out a message to cluster head to ask for moving out, and the message is created in accordance with the following equation. Upon receiving such a message, cluster head is supposed to delete information on the outgoing node.

$$\text{OutMsg} := \{ \text{Node}(N), \text{PubKey}(\text{CH}), \text{Validity}(t), \text{"Out"} \}$$

### 2.4 Combination of Clusters

When a cluster is integrated into another cluster, the cost is high and network overhead significantly increased. If there are two clusters with distance of 1-hop, they need to be combined. The role of cluster head is continually assumed by either of two cluster heads which had a larger number of member nodes and higher trust value. And nodes within the cluster whose head quit are required to receive a certificate again from the new cluster head. And the former cluster head transmit information on its member nodes to the new cluster head.

## 3 Analysis

The performance of the proposed data transmission solution was evaluated using the ns2 simulator. The efficiency of ad hoc network with the proposed solution is measured by the development of path, overhead of cluster formation and the amount of control packet.

### 3.1 Scenario for Evaluation

The mobility models of mobile nodes were limited 2D models in which the direction of movement was not restricted under the assumption that users are evenly distributed in the ad hoc network and that users are activated in open-ended environment rather than restricted environment.

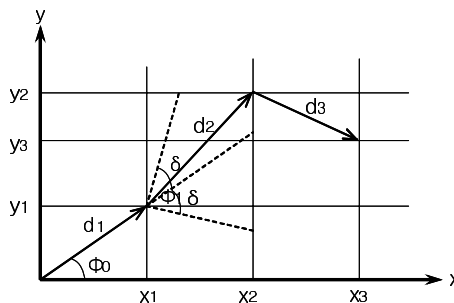


Fig. 1. Direction of movement of nodes

Fig. 1 showed the direction of movement of mobile nodes in the beginning. It was found that nodes took a new direction after following the preset direction at a present speed for the preset time duration.

### 3.2 Results of Experiment

The size of packet was 64 bites and pause time was 25 seconds during experiment. The transmission range was 250m. Experiments were conducted 10 times for each of differently sized networks for 300 seconds each time. The average number of cluster heads was counted to evaluate the stability of network and clusters and similarity between clusters. As shown in Fig. 2, the number of cluster heads remained steady regardless of changes in the number of nodes within the network and movement speed. The number of cluster heads, which varies with transmission radius of nodes, accounted for 21% of total nodes on average in each experiment. In the proposed model, cluster head receives the trust value of a given node from neighboring nodes. In case cluster head receives the trust value of the same node from different nodes, an

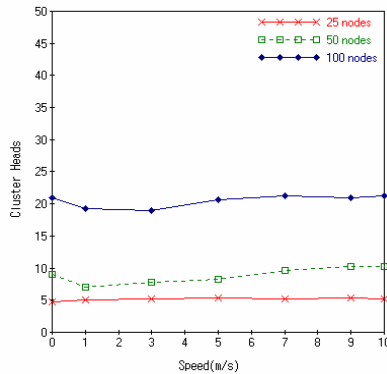


Fig. 2. Changes in the number of cluster heads with movement speed

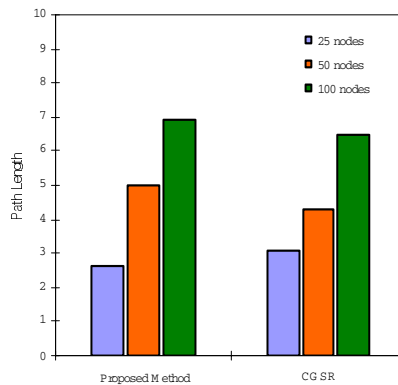


Fig. 3. Path distance

average trust value is calculated and assigned to the node. The trust value of nodes is essential in the proposed concept. When a cluster head receives RREQ packet, it evaluates the trust of neighboring nodes to transfer RREQ packet to its neighbors, which forward the packet to their neighbors and so on until the destination was found. If a short path is discovered among nodes with low trust ratings, cluster head will retransmit RREQ packet to establish a new path with nodes with high trust ratings, increasing the safety in data transmission. Given the mechanism that the value of trust for a node is related to its experience with other nodes, the interaction between nodes increases while the number of selfish nodes decreases. Fig. 3 shows the average length of path. The average length of path was a bit longer in the 50 nodes network and 100 nodes network, compared with those of other solutions. The difference is due to the mechanism of the proposed solution in which nodes are linked based on their trust values rather than close distance.

The average delay time of packet between source node and destination node is shown in Fig. 4. There was a correlation between the average delay time and path length, and this correlation is explained by the trust value-based path creation described in the previous paragraph.

Changes of the amount of control packet in the 100 nodes random network are shown in Fig. 5. The amount of control packet has a significant impact on the performance of network because of low bandwidth on wireless networks. A limited broadcast mechanism was used to reduce the amount of control packet. It was found during experiment that the number of path adjustments was relatively low thanks to high trust ratings of intermediate nodes.

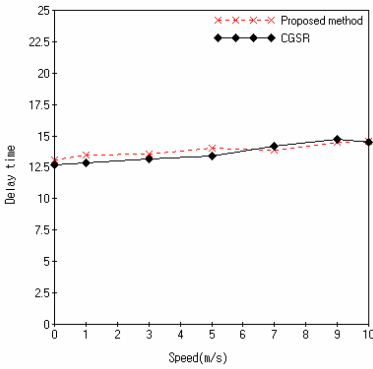


Fig. 4. Average delay time

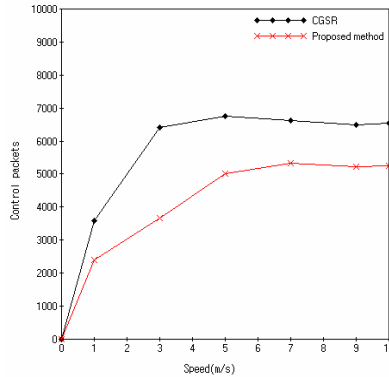


Fig. 5. Changes of the amount of control packet

## 4 Conclusion

This study proposed a cluster-based data transmission solution for ad hoc networks. It is important for a data transmission solution to meet the requirements associated with dynamic topology and various types of node linkage in ad hoc networks. When a cluster head is elected, neighboring nodes transfer the trust values of their neighbors to cluster head, which is able to evaluate and guarantee the trust of each node without



its own experience with each node. And a limited broadcast mechanism was used to enhance efficiency in the network by reducing the amount of control packet. The proposed model provided a high degree of stability and security in the ad hoc network. Further study is needed to develop an optimum algorithm for cluster formation and investigate power consumption of nodes.

## References

1. H. Luo, S. Lu, "Ubiquitous and Robust Authentication Services for Ad Hoc Wireless Networks", Technical Report TR-200030, Dept. of Computer Science, UCLA, 2000.
2. L. Venkatraman and D.P. Agrawal, "A Novel Authentication scheme for Ad Hoc Networks", Wireless Communications and Networking Conference, 2000.
3. Y. Desmedt and S. Jajodia, "Redistributing secret shares to new access structures and its applications", George Mason Univ., Tech. Rep., 1997.
4. L. Zhou and Z. J. Hass, "Securing ad hoc networks", IEEE Network, vol. 13, no. 6, pp.24-30, 1999.
5. A. Bayya, S. Gupte, Y. Shukla, and A. Garikapati, "Security in Ad hoc networks", CS 685, Computer Science Department University of Kentucky.
6. H. Lo, P. Zerfos, J. Kong, S. Lu, and L. Zhang, "Self-securing ad hoc wireless networks", in Proc. 7<sup>th</sup> IEEE Symp. On Comp. and Communication (ISCC), Taormina, 2002.

# Embodied Conversational Agent Based on Semantic Web

Mikako Kimura and Yasuhiko Kitamura

Department of Informatics  
Kwansei Gakuin University  
2-1 Gakuin, Sanda, Hyogo 669-1337, Japan  
{mikako.kimura, ykitamura}@ksc.kwansei.ac.jp

**Abstract.** Embodied conversational agents (ECA's) are cartoon-like characters which interact with users through conversation and gestures on a computer screen. ECA makes human computer interactions more friendly because we can use most human-like communication skills such as natural conversation. ECA's are useful as Web guides by incorporating them into Web browsers. They guide us around Web pages chatting with us. To build such an agent, we need to describe a scenario to explain Web pages. Conventionally such scenarios are written manually by developers or programmers using a dialogue description language such as AIML (Artificial Intelligence Markup Language), so it is difficult to update them when Web pages are updated. In this paper, we propose a scheme to automatically generate utterances of Web guide agents depending on Web pages. To this end, we need to make agents understand the contents of Web pages and to make them talk according to the contents, so we utilize RDF (Resource Description Framework) to present the semantic contents of Web pages. To make agents talk according to the contents, we utilize a RDF query language SPARQL (Simple Protocol And RDF Query Language) and extend the AIML language to incorporate SPARQL query in it. As a prototype, we developed a Web guide system employing an ECA.

**Keywords:** Agents and Semantic Web, Embodied conversational agents, RDF, SPARQL, AIML.

## 1 Introduction

Embodied conversational agents (ECA's)<sup>1</sup> are cartoon-like characters which interact with users through conversation and gestures on a computer screen [4]. ECA makes human computer interactions more friendly because we can use most human-like communication skills such as natural conversation, and releases us from using machine-dependent communication methods like commands or programs.

ECA's are applied to many fields such as entertainment, story telling, education, e-commerce, community support, and so on by utilizing verbal and non-verbal interfaces [9]. ECA's are also useful as Web guides by incorporating agents into Web browsers. They guide us around Web pages with chatting. To build such an agent, we

---

<sup>1</sup> ECA's are called life-like characters [9], animated characters [8], synthetic characters [6], or believable agents [3].

need to describe a scenario to explain Web pages. AIML is an XML language to describe conversation scenarios for ECA's [10]. It specifies pairs of a pattern and a template. When an utterance from a user matches a pattern in the pair, the agent talks a phrase in the template of the pair. Conventionally such scenarios are written manually by developers or programmers considering contents of Web pages, so it is difficult to update them when the Web pages are updated.

In this paper, we propose a scheme to automatically generate utterances of Web guide agents. To this end, we need to make agents understand the contents of Web pages and to make them talk according to them, so we utilize RDF (Resource Description Framework)<sup>2</sup> to represent the semantic contents of Web pages. To make agents talk according to the contents, we utilize a RDF query language SPARQL (Simple Protocol And RDF Query Language)<sup>3</sup> and extend the AIML language to incorporate SPARQL query in it.

In Section 2, we discuss ECA's, their related work, and AIML. In Section 3, we discuss how to integrate RDF, SPARQL, and the extended AIML to automatically generate utterances of Web agents. In Section 4, we show an interactive Web guide system as an implementation of an ECA based on Semantic Web. We conclude this paper in Section 5 with our future work.

## 2 Embodied Conversational Agents

ECA's are virtual animals or human beings which behave like real ones on a computer screen interacting with users through conversations and gestures. Interactions between human users and computers become more friendly by using ECA's as their interface.

Microsoft Agents [5,2] are an example of ECA's and run on Microsoft Windows machines. They can recognize utterances from users by using a speech recognition module and reply to them with gestures by a speech synthesis module. We are not sure that Microsoft Agents have made a real success but they are well known as a helper in Microsoft Office. By using Visual Basic or JavaScript, they can be deployed as a user-friendly interface of a number of applications on Windows. Other commercial products can be found on the Web like Extempo<sup>4</sup>, Haptek<sup>5</sup>, Verbots<sup>6</sup>, and Artificial Life<sup>7</sup>.

We can employ ECA's as Web guide agents. Such a Web guide navigates a user to Web pages that may be interesting to him/her with chatting. It makes a Web site more attractive and user-friendly especially for novice Internet/computer users. InfoWiz<sup>8</sup> developed at SRI International and AiA personas [1] developed at DFKI are examples of early systems.

ALICE (Artificial Linguistic Internet Computer Entity)<sup>9</sup> is a chat bot using AIML (Artificial Intelligence Markup Language) [10], which is an XML based language to

<sup>2</sup> <http://www.w3.org/RDF/>

<sup>3</sup> <http://www.w3.org/TR/rdf-sparql-query/>

<sup>4</sup> <http://www.extempo.com>

<sup>5</sup> <http://www.haptek.com>

<sup>6</sup> <http://www.verbots.com/>

<sup>7</sup> <http://www.artificial-life.com>

<sup>8</sup> <http://www.ai.sri.com/oaa/infowiz.html>

<sup>9</sup> <http://www.alicebot.org/>

```

<aiml>
  <category>
    <pattern>How are you?</pattern>
    <template>I am fine, thank you.</template>
  </category>
</aiml>

```

**Fig. 1.** An example of AIML category

describe knowledge of an agent. The basic unit of knowledge is called a category and an AIML category consists of two elements: one pattern and one template as shown in Fig. 1. The pattern is an input from a user and the template is a response from the agent. As in Fig. 1, if a user says "How are you?" to the agent, it replies "I am fine, thank you."

Pandrabot<sup>10</sup> is a Web-based chat bot hosting service. On a Web browser, we can build our own chat bot by using AIML and make it accessible through the Internet. By using Pandrabot, we can build a Web guide easily. At present, programmers/developers manually create conversational rules in AIML considering the contents of Web pages. If a Web page is updated, however, rules related to the page have to be updated. When a number of Web pages are updated frequently, the task to update the conversational rules becomes cumbersome.

In this paper, we propose a scheme to automatically generate utterances depending on Web pages by incorporating the Semantic Web technology into AIML. We utilize the RDF to make Web guide agents understand the contents of Web pages and a RDF query language SPARQL to extract requested information, which is specified by the user, from the RDF data. We extend AIML as it can handle SPARQL queries and can generate utterances which contain the requested results.

### 3 Generating Dialogue Based on Semantic Web

We propose a scheme to automatically generate utterances based on the Semantic Web technologies. To achieve this goal, the agent needs (1) to understand the contents of Web pages and (2) to talk to users according to them. To achieve (1), we use the RDF to represent the contents of Web pages in a machine-understandable way and a RDF query language SPARQL to extract requested one from RDF data. To achieve (2), we extend the AIML as it contains a SPARQL query.

#### 3.1 RDF

RDF (Resource Description Framework) is a framework to describe meta data. By using RDF, we can describe the meaning of Web pages in a machine-understandable way. RDF uses triples to describe logical relations. A triple consists of a resource, a property, and a value. The meaning of the triple is that the property of the resource is represented as the value.

<sup>10</sup> <http://www.pandorabots.com/botmaster/en/home>

```

<?xml version="1.0" ?>
<rdf:RDF
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:foaf="http://xmlns.com/foaf/0.1/"
  xmlns:dc="http://purl.org/dc/elements/1.1/@"
  . . . . .
  <foaf:Person rdf:ID="me">
    <foaf:family_name>Kimura</foaf:family_name>
    <foaf:firstName>Mikako</foaf:firstName>
    <foaf:gender>female</foaf:gender>
    <foaf:mbox>mikako.kimura@ksc.kwansei.ac.jp</foaf:mbox>
    <foaf:currentProject>
      <foaf:Project>
        <dc:title>Web information integration using character agents
        </dc:title>
        <dc:description>This project develops a character agent which
        guides Web pages interacting with users.</dc:description>
        </foaf:Project>
      </foaf:currentProject>
    </foaf:Person>
  . . . . .
</rdf:RDF>

```

**Fig. 2.** RDF data

We show an example of RDF data in Fig. 2. In this example, we use the FOAF<sup>11</sup> ontology and the Dublin Core (DC)<sup>12</sup> ontology to describe laboratory information. The FOAF (Friend Of A Friend) project provides an ontology to describe people and their relations. Information of a person is written by using a class `foaf:Person` with properties such as names, gender, and projects. In the above example, a FOAF data concerning Mikako Kimura is shown.

DC ontology is another ontology that specifies metadata of Web resources. The FOAF ontology is not enough to describe laboratory information and the DC ontology complements it. In the above example, we use the DC ontology to describe the title and the description of project.

### 3.2 SPARQL

Jena<sup>13</sup> is a Java class library to handle RDF data and provides a RDF query language SPARQL (Simple Protocol And RDF Query Language). By using SPARQL, we can extract requested one from RDF data by submitting a SPARQL query. Fig. 3 shows an example of SPARQL query. This query returns a mail address, specified by `foaf:mbox`, of Mikako Kimura, specified by `foaf:name`. In the query, a variable `?x` is used to designate the FOAF node of Mikako Kimura. If we apply this query to the RDF data in Fig. 2, a reply "mikako.kimura@ksc.kwansei.ac.jp" returns.

<sup>11</sup> <http://www.foaf-project.org/>

<sup>12</sup> <http://dublincore.org/>

<sup>13</sup> <http://jena.sourceforge.net/>

```

PREFIX foaf: <http://xmlns.com/foaf/0.1/>
SELECT ?mbox
WHERE
{
  ?x foaf:name "Mikako Kimura" .
  ?x foaf:mbox ?mbox
}

```

**Fig. 3.** An example of SPARQL query

### 3.3 Extended AIML

As mentioned in Section 2, AIML supports only fixed conversational rules between an agent and a user, so we need to extend it to make it reflect the corresponding Web updates by incorporating a SPARQL query in it.

As shown in Fig. 4, we insert `<sparql>` and `<url>` tags into the AIML specification. We also modify descriptions in `<pattern>` and `<template>` tags as we can handle variables. For example, if a user submits a query "What is Kimura's current project?", the query matches a pattern in the AIML description and variable `$x` is set to be "Kimura". Then, the system executes a SPARQL query specified in `<sparql>` tags. As `$x` is set to "Kimura", the query retrieves the title of Kimura's current project from a RDF file named "member.rdf" (shown in Fig. 2) and the result is set to variable `?ans`. Finally, the system replies to the user saying "Kimura's current project is Web information integration using character agents."

`<url>` tags are used to show a Web page whose URL is specified between the tags on a browser.

```

<aiml>
....
<category>
  <pattern>What is $x's current project?</pattern>
  <template>$x's current project is ?ans.</template>
  <sparql>
    PREFIX rdf: &lt;http://www.w3.org/1999/02/22-rdf-syntax-ns#&gt;
    &gt;
    PREFIX foaf: &lt;http://xmlns.com/foaf/0.1/&gt;
    PREFIX dc: &lt;http://purl.org/dc/elements/1.1/&gt;
    SELECT ?ans
    FROM NAMED <member.rdf>
    WHERE {
      ?a foaf:family_name $x .
      ?a foaf:currentProject ?y .
      ?y dc:title ?ans .
    }
  </sparql>
  <URL>http://ist.ksc.kwansei.ac.jp/~kitamura/lab/projects-j.htm
  </URL>
</category>
....
</aiml>

```

**Fig. 4.** An example of extended AIML

### 4 Interactive Web Guide Agent

As a prototype of ECA based on Semantic Web, we developed an interactive Web guide agent for our laboratory.

Its system architecture is shown in Fig. 5. The system navigates a user who have interest in Web pages of Kitamura laboratory. The contents of the Web pages is described in a RDF file.

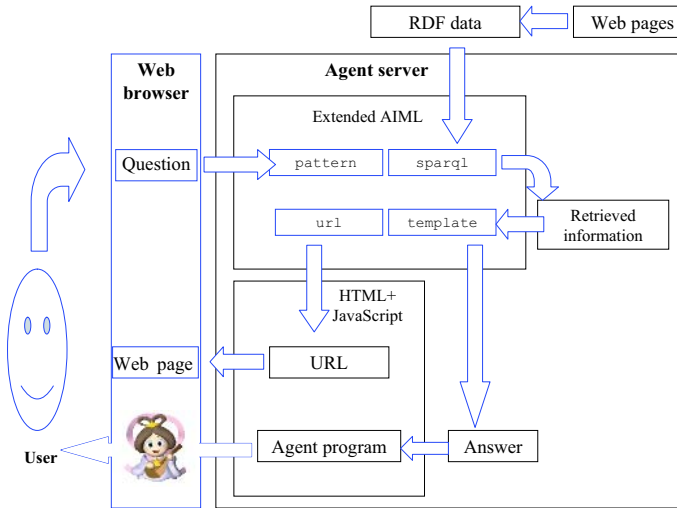


Fig. 5. System architecture of interactive laboratory guide agent

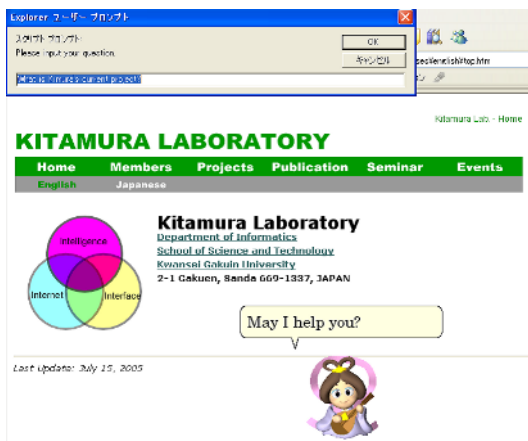


Fig. 6. Query submission interacting with ECA

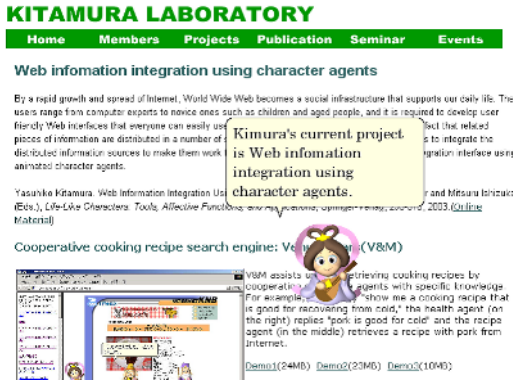


Fig. 7. Reply from ECA

The system has an ECA as its interface to receive a question from a user as shown in Fig. 6. We use the Microsoft Agents platform and JavaScript to implement the ECA. It first shows a dialogue box saying "May I help you?". To the dialogue box, a user submits a query such as "What is Kimura's current project?".

The question submitted to the dialogue box is sent to its Web server, which is implemented by Tomcat. The server searches its AIML file for a conversational rule that matches the question. If a rule is found, the server submits a SPARQL query in the rule and extracts data from the designated RDF file. It then inserts the extracted data into the template in the rule. Finally the server returns a HTML file with JavaScript. The browser shows a Web page specified by `<url>` tags and the attached ECA replies an answer as shown in Fig. 7.

## 5 Summary and Future Work

We have developed a prototype of ECA based on Semantic Web. The ECA receives a question about Web pages from a user and replies with an answer, which reflects the contents of the Web page, to him/her. The contents of Web pages are represented in RDF. When the RDF data is updated, the answer changes depending on the update.

The conversation skill of ECA highly depends on the number of conversational rules. To make an ECA fluently interact with a user, we need to provide a large number of rules. On the other hand, it is very tough to update them according to the update of Web pages. Our proposed scheme alleviates the developer/programmer from the maintaining task of conversational rules.

At present, the ability of Web guide agent to answer questions depends on the specification of SPARQL, so it is not easy to answer a question like "How many members does Kitamura lab have?", because the current SPARQL cannot handle aggregate functions such as COUNT() in SQL [7]. How to deal with this problem remains as our future work.

Another future work relates to the inference mechanism. At present, our extended AIML handles only a simple form of SPARQL, so we can enhance the function by



adding an inference mechanism if we can combine multiple SPARQL queries in AIML. For example, an ECA may be able to use the right title of person properly from the RDF data. If a person is male, then the ECA uses the title “Mr.” and if female, it uses “Mrs.” or “Miss” depending on her marriage record. ECA may call him/her “Professor” referring to his/her occupation data.

## References

1. Elisabetu André, Thomas Rist, and Jochem Muller. Employing ai methods to control the behavior of animated interface agents. *Applied Artificial Intelligence*, 13:415–448, 1999.
2. Gene Ball, Dan Ling, David Kurlander, John Miller, David Pugh, Tim Skelly, Andy Stankosky, David Thiel, Maarten Van Danzich, and Trace Wax. Lifelike computer characters: The persona project at microsoft. In Jeffrey M. Bradshaw, editor, *Software Agents*, pages 191–222. AAA Press/The MIT Press, 1997.
3. Joseph Batesi. The role of emotion in believable agents. *Communications of the ACM*, 37(7):122–125, July 1994.
4. Justine Cassell, Joseph Sullivann, Scott Prevost, and Elizabeth Churchill, editors. *Embodied Conversational Agents*. MIT Press, 2000.
5. Microsoft Corporation. *Microsoft Agent Software Development Kit*. Microsoft Press, 1999.
6. Clark Elliott and Jacek Brzezinski. Autonomous agents as synthetic characters. *AI Magazine*, 19(2):13–30, 1998.
7. Hector Garcia-Molina, Jeffrey D. Ullman, and Jennifer Widom. *Database Systems: The Complete Book*. Prentice Hall, 2002.
8. Barbara Hayes-Roth. Animated characters. *Autonomous Agents and Multi-Agent Systems*, 1:195–230, 1998.
9. Helmut Prendinger and Mitsuru Ishizuka, editors. *Life-Like Characters: Tools, Affective Functions, and Applications*. Springer-Verlag, 2004.
10. Richard Wallace. *The Elements of AIML Style*. ALICE A. I. FoundationCInc., 2003.

# Dynamic Service Composition Model for Ubiquitous Service Environments

Seungkeun Lee and Junghyun Lee

School of Computer Sci. & Eng., Inha Univ. Incheon, South Korea  
{sglee, jhlee}@nlsun.inha.ac.kr

**Abstract.** There are a lot of services in ubiquitous computing environments. So, ubiquitous service need a service matchmaker which presents more easily and with accuracy. The coupling of webservices and semantic web technology provides the ability to automatically discover, compose and execute webservices. However, the composition of services is generally static because these services are usually described using BPEL4WS or WSFL, restricting dynamic operation because the composite service only has a sequence execution plan. This dynamic composition cannot generate a parallel execution plan for many Internet business applications. In this paper, we design an ontology based framework for dynamic webservice composition. Also, we present a semantic webservice framework using dynamic composition model. This dynamic composition model can generate a parallel execution plan. These plans are calculated using QoS model, hence the best execution plan is selected.

## 1 Introduction

In ubiquitous computing environment, service can be located in many device and service gateway. So, the precisely service matchmaker is need to service management in ubiquitous environments[1]. In many research, webservices technology is used in service development and management in ubiquitous computing environment. A webservice is a core technology in e-Business and are researched by a number of people by a spread using of XML(eXtensible Markup Language), WSDL(Web Service Definition Language), SOAP(Simple Object Access Protocol). However, because a webservice is composed only by syntactic information described by in XML, the structure cannot process semantic of contents. Due to this issue, efforts toward the adaptation of the Semantic Web to webservices are gaining momentum[2]. This enables webservices to be accessed by contents rather than by keywords. Webservices can be discovered, selected and composed automatically by other services.

Services can be divided into two components, a simple service and a composition service. A simple service is an Internet application which is independent of other services, to achieve customer satisfaction. Examples of simple service are a hotel reservation services and booking service for an airplane. A composition service is composed by some webservices for satisfaction their customers. An Example of a composition service is a touring reservation service. This is composed of a hotel reservation service and airplane booking service, which individually, can be simple services. This composition is very important because it presents approaches for all kinds

of WWW activities. However, many researches are focused on static composition using BPEL4WS(Business Process Execution Language) or WSFL(Web Service Flow Language)[3]. A static composition has problems in a dynamic WWW environment. If a webservice is modified once, all composition plans having the modified webservice must be redesigned. It is largely an ad-hoc, time-consuming and error prone process. The dynamic composition of webservices is noticed by many researchers [4]. These researchers' studies have not been completed yet and therefore suitability in the diversity of a business environment cannot be determined. Dynamic composition generates an execution plan with webservices, IOPE(Input/Output/ Pre-condition/Effect). This execution plan is a sequence list of executions of simple webservices. For examples, a touring reservation received parameter, made of tour duration and place information used to call the hotel reservation service and booking service for an airplane. But, there is no relationship between the Input/output two webservice, so an execution plan including two services cannot be made dynamically. These services are executed in parallel and are composed as a composition service[5].

This paper presents an ontology-based framework for dynamic composition of services. This ontology is designed by an extension of OWL-S[6,7]. A presented framework can generate an execution plan having a sequence or parallel execution plan of webservices automatically. Also, this framework estimates the some execution plans automatically using the QoS model and selecting the best execution plan. A selected plan is translated into an OWL-S ServiceProcess model. This framework has some benefits which are not described previously. (1) It can create an execution plan having a sequence plan and a parallel plan. (2) It can select the best execution plan, using the QoS model presented by the designed ontology. (4) A selected execution plan is used in various execution environments, supporting OWL-S.

## 2 Dynamic Composition Model Using Ontology

The dynamic composition of semantic web services creates proper execution plans of composite webservices and selects a best execution plan using the QoS properties of a composite webservices. In this section, we propose the ontology for execution plan creation, and the evaluation of an execution plan for best execution plan selection. It is based on extensions of OWL-S.

### 2.1 Ontology Model

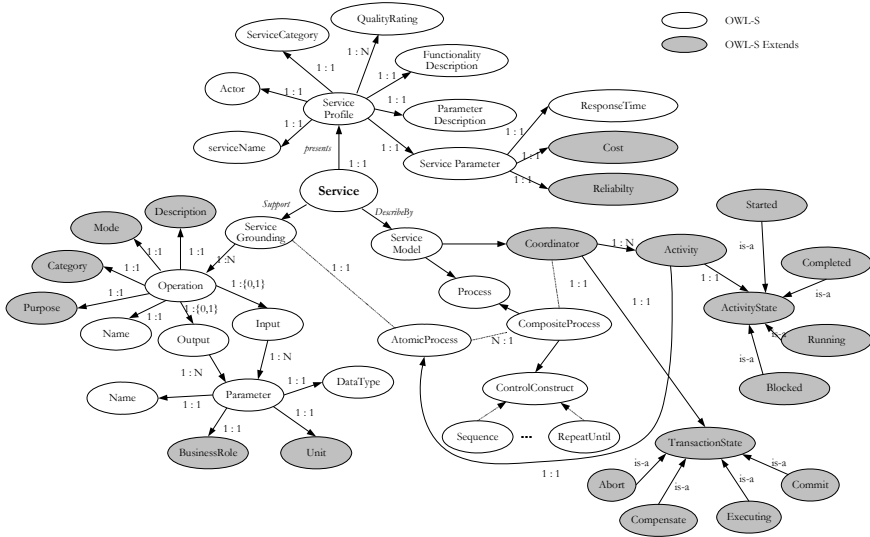
In this paper, we designed a ontology model, the extends of OWL-S for this framework. Fig describes this ontology model.

The QoS ontology is composed of three properties. These are a response time, a execution cost, and the reliability of webservice. Each property has three values, min/average/max. A response time describes the time from calling an operation to getting a response from a webservice. An execution cost describes the total cost in terms of resources utilization. Reliability describes the rate in which webservices are executed correctly.

A connection between webservices is achieved by exchanging message. A message describes datatype, name, unit and role. The business role gives the semantics of the

corresponding parameter. It takes its value from a predefined taxonomy for business roles. In order to connect between two webservices the mapping between messages of two webservices must be completed.

The functionalities provided by a webservice are accessible through operation invocations. We consider four operation modes. ‘one-way’, ‘notification’, ‘solicit-response’, and ‘request-response’. This is described in detail, in Table 1.



**Fig. 1.** The Extends of OWL-S Ontology

**Table 1.** Operation mode ontology

Operation mode	Description
one-way	Input message, No output message. A pair with notification
Notification	No Input message, Output message. A pair with one-way
Solicit-response	Input – Output. A pair with request-response
request-response	Output – Input. A pair with solicit-response

A composite service is treated as a logical webservice. This service is executed using transaction methods. We designed properties for a transaction. A state of transaction is defined with ‘commit’, ‘executing’, ‘compensate’, and ‘abort’ properties. An activity is defined with ‘started’, ‘completed’, ‘running’, and ‘blocked’. This ontology is not within the scope in this paper. It will be published in a subsequent papers.

**2.2 Webservice Composability**

In this section, we describe how the framework can decide whether two webservices can be composed together. A composition of two webservices is achieved by an operation of

one webservice call to an operation of another webservice. An calling of operation call is achieved by a sending a message. In order to decide whether two webservices can be composed, message compatibility and operation compatibility are verified.

**Message Compatibility.** Interoperation is achieved by exchanging messages. A message could be composed of parameters, having specific datatypes. A sending parameter must be interoperable with a receiving parameter. Every parameter would have well-defined semantics according to that taxonomy. We consider two primary datatype-compatibility methods: direct and indirect compatibility. Two parameters are directly compatible if they have the same data type. A parameter  $p$  is indirectly compatible with a  $q$  if the type of  $p$  is derived from the type  $q$ . We extend the notion of data type compatibility to messages as follows: A message  $M$  is a datatype compatible with a message  $N$  if every parameter of  $M$  is directly or indirectly compatible with a parameter of  $N$ . Note that not all parameters of  $N$  need to be mapped to the parameters of  $N$ . This algorithm describes deciding of message compatibility. In this algorithm,  $P_i$  is a parameter name included in message.  $T$  is the datatype,  $U$  is the unit, and  $R$  is a business role.

**Algorithm 1.** message\_compatibility

```

Input :  $M_i = \{P_i, T_i, U_i, R_i\}$ ,  $M_j = \{P_j, T_j, U_j, R_j\}$ 
Output : True / False
Begin message_compoable
  matchedList = false;
  for eache param  $p_{ik} \in P_i$  do
    found = false
    for each param  $p_{jl} \in P_j$  |  $p_{jl}$  is not matchedList do
      if ( $T(p_{ik}) = T(p_{jl})$  or  $T(p_{jl})$  is derived from  $T(p_{ik})$ )
and ( $U(p_{ik}) = U(p_{jl})$ ) and
      ( $R(p_{ik}) = R(p_{jl})$ ) then
        found = true;
        matched = matched  $\cup$   $p_{jl}$ ;
      end if
      if found is not true then return false;
    end for
  return true;
end for
End.
```

**Operation Compatibility.** In order to be compatible with other operations, these operations have a “dual” mode. As described in Table 1. For example, if an operation mode of one webservice is “one-way” then the other operation mode is “notification”. Also, two operations have the same purpose properties and category properties. A detailed algorithm is omitted because of a restricted coverage in this paper.

### 2.3 Automatic Creation of Execution Plan

In this section, we describe how an execution plan can be created automatically. First, chainStartList is generated from user requirements. If a webservice can satisfy a

user’s requirements, there is one element within chainStartList. If a webservice can satisfy a user’s combined requirement, the some webservice will be elements of chainStartList. The chainStartList variables, is used when composing a sequence of webservices. If inputs of a webservice, added to a chain, are a subset of user’s inputs(weHave) then webservice composition in this chain is stopped. Finally, all chains are entered for execution plan of composite webservices Algorithm 2 describes the creation of an automatic execution plan.

**2.4 Automatic Creation of Execution Plan**

The A composition of webservices can be treated as a long-termed transaction. It is important that the optimum execution plan is selected. In order to select a best he optimum execution plan, we use QoS properties, response time, cost and reliability. An execution plan with a large QtValue(S)is selected by a best execution plan. A QtValue(S) is calculated by follows

$$\begin{aligned}
 Reliability &= (1 - \frac{failure}{totalExecution}) * 100(\%) \\
 QtValue(S) &= \frac{m3 * dim(S, Reliability)}{m1 * dim(S, Time) * m2 * dim(S, Cost)} \\
 dim(S, dim) &= \sqrt[3]{Min(S, dim) * Avr(S, dim) * Max(S, dim)}
 \end{aligned}$$

If a web service is a composite webservice the following is used for a calculation of QtValue(S).

$$\begin{aligned}
 dim(CompositeS, Time) &= \sum_{i=1} dim(Si, Time) \quad \dots \text{sequence execution plan} \\
 dim(CompositeS, Time) &= Max[dim(Si, Time)] \quad \dots \text{parallel execution plan} \\
 dim(CompositeS, Cost) &= \sum_{i=1} dim(Si, Cost) \\
 dim(CompositeS, Cost) &= \prod_{i=1} dim(Si, Reliability)
 \end{aligned}$$

**3 Conclusion**

We designed the ontology based service composition for ubiquitous computing environment. This dynamic composition of webservices and a model for dynamic composition and calculation for finding the best execution plan. This model has these merit. It can generate a best execution plan dynamically using ontology and QoS properties. It can be adapted to a variety of business applications. And, This plan can be translated to OWL-S, for usage in any execution environment supporting OWL-S. In the future, we will extend this framework using a transaction model. This will achieve execution, monitoring, and fault handling of composite services.

## Acknowledgement

This research was supported by the MIC(Ministry of Information and Communication), Korea, under the ITRC(Information Technology Research Center) support program supervised by the IITA(Institute of Information Technology Assessment) (IITA-2005-C1090-0502-0031).

## References

1. S. Lee et al, "Service Mobility Manager for OSGi Framework," Computational Science and Its Application 2006, LNCS3983, Springer, 2006.
2. T. Berners-Lee, J. Hendler, O.Lassila, The Semantic Web. Scientific American, 284(5):34-43. 2001
3. D. Fensel, C. Bussler, "The Web Service Modeling Framwork WSMF", Whit Paper and International Report, [www.cs.vu.nl/swws/download/wsmf.paper.pdf](http://www.cs.vu.nl/swws/download/wsmf.paper.pdf), 2002
4. M. Paolucci, K. Sycara, "Autonomous Semantic Web Services", IEEE Computer Society, 2003.
5. D. J. Mandell, S. A. McIlraith., "A Botton-Up Approache to Automating Web Service Discovery, Customization and Semantic Web Translation, KSL Lab, Stanford University, 2003.
6. RDF Primer, <http://www.w3.org/TR/rdf-primer/>, 2004
7. OWL-S, <http://www.w3.org/Submission/2004/SUBM-OWL-S-20041122/>, 2004

# Framework for Agent-Based Buying Decision Process

Sazalinsyah Razali and Mashanum Osman

Faculty of Information & Communication Technology, Kolej Universiti Teknikal  
Kebangsaan Malaysia  
75450 Melaka, Malaysia  
{sazalinsyah, mashanum}@kutkm.edu.my

**Abstract.** In traditional business model, the buying decision process is poorly coordinated among the human decision-makers. Therefore, a long-lived, adaptive, and autonomous application called software agents, that can perform tasks such as personalization, brokering, and negotiation in e-commerce is much needed. These applications reside at the buyers' side or at the sellers' servers. The purpose of this paper is to research into possible deployment of software agents in a framework for e-commerce buying decision process. This paper overviews the traditional business model, the Consumer Buying Behavior (CBB) model, and also covers the requirements needed for minimizing human interactions in buying decision processes. The research proposes a software agent's framework in which, two main approaches, namely Automated Collaborative Filtering (ACF) and Better Business Bureau (BBB), are merged to produce better agents in assisting buying decision process. The framework will enable the agents to get the best price for a good product from a reputable merchant.

## 1 Introduction

Two general purpose of e-commerce is interoperation and automation, whereby in most cases there is a dependency of automation upon interoperation [4]. Systems developed to cater the e-commerce process are using different platforms, standards, representations, and languages. However, there is an increasing need for these systems to interoperate; communicating information from one system to another; to have a better and efficient business environment. Heterogeneity poses great difficulties in realizing interoperation [5]. In order to resolve this issue, software agents are deployed to automate many steps in the e-commerce process, which in turn can minimize cost and consequently maximizing profit.

Currently, there is no one standard, universally accepted definition of software agent [6]. However, generally software agents are programs that are long-lived, adaptive, autonomous, goal-oriented, collaborative, proactive, and mobile [6]. Another important aspect is that software agents are entities that carry out some set of operations or tasks on behalf of a user or another program with some degree of independence or autonomy. Software agents have diverse roles and opportunities in e-commerce, from assisting in product searching, merchant identifying, negotiation, payments, after-sales support, recommendation systems, automating supply chain management, and numerous other back-office tasks. From the Consumer Buying



Behavior (CBB) model that will be explained in Section 3, agents can be seen playing significant roles in three primary CBB stages; Product Brokering, Merchant Brokering, and Negotiation corresponding to what to buy, who to buy it from, and how to determine the terms of the transaction respectively [4].

## 2 Overview of Traditional Business Model

In traditional business as well as in e-commerce, firms or organizations engage in many other activities besides buying or selling that keep them in business, such as promoting or advertising their products or services, identify demand, deliver its products or services, and provide after-sales support. It is a negotiated exchange of products or services between two parties or more and includes all activities that each of the parties undertakes to complete the transaction [7].

The essence in businesses is the buying process whereby the steps of product searching, vendor selection and transaction negotiation are vital. These three steps can be regarded as the buying decision process in business. However, in e-commerce transactions, this process is quite complicated and difficult to execute because of human decision-makers limitations. A human can only work in a specified time period, and the various kinds of products and services required by a company may result in negotiation with not one seller, but many sellers at a time. These and many other limitations make the buying decision process poorly coordinated among the human decision-makers.

## 3 Buying Decision Process

This section will look into the buying decision process as well as how software agents can minimize human involvement in this process.

### 3.1 Consumer Buying Behavior (CBB) Models

There are several descriptive theories and models that attempt to capture consumer-buying behavior. The model presented here is called the Consumer Buying Behavior (CBB) Model [9] [1].

The first stage in the CBB model is where the consumer becoming aware of some unmet need. At this stage the consumer can be stimulated through product information. The Product Brokering stage comprises the retrieval of information to help determine what to buy which also covers evaluation of product alternatives. The result of this stage is the 'consideration set' of products. This 'consideration set' combine with merchant-specific information is used to determine who to buy from. This stage includes the evaluation of merchant alternatives based on consumer-provided criteria, example price, warranty, availability, delivery time, and reputation.

The fourth stage in the CBB model is the negotiation stage, which is about how to determine the terms of the transaction. Negotiation varies in duration and complexity depending on the market. Purchase and delivery of a product can either signal the termination of the negotiation stage or occur sometime afterwards, in either order. In some cases, the payment methods available and delivery options can influence product

and merchant brokering stage. The last stage is the product service and evaluation that involves product service, customer service, and an evaluation of the satisfaction of the overall buying experience and decision. These stages in the CBB model represent an approximation and simplification of complex behaviors [1]. Furthermore, these stages often overlap and migration from one to another can be non-linear and iterative.

### **3.2 Minimizing Human Interactions**

One of the greatest roles of software agents is minimizing human interactions, especially in the buying decision process. It can automate much of the tasks in the process of identification, selection, negotiation, and purchasing. An example is negotiation done by software agents, which is defined as the process by which group of agents communicate with one another to try to come to a mutually acceptable agreement on some matter [2]. Nevertheless, there are many requirements that needed to be fulfilled for software agents in order to minimize human interactions in these processes. Some of these significant requirements are security, interoperability, and acceptance. The agent owners must be assured that the agent will not compromise private information, which includes account numbers for payment mechanisms, and deviate beyond its constraints [4]. In addition, agent owners must trust the environment in which agents carry out their tasks. Software agents should also be capable of operating in different environments or platforms, because it needs to communicate with other agents that may be on a different platform. These requirements fulfillment can realize the notable role of software agents in minimizing human interactions in general.

## **4 Agent's Buying Decision Process**

There are already many software agents that have been developed either for commercial or academic purposes that can perform tasks in one or more stages in the CBB model. This section will give an overview on two approaches used by the available software agents.

### **4.1 Automated Collaborative Filtering (ACF)**

This approach works in the product brokering stage of the CBB model. It uses a "word of mouth" recommendation mechanism or aptly named automated collaborative filtering (ACF). In essence, the approach uses the opinions of like-minded people to offer recommendations. One example software agent that utilizes the ACF approach is Firefly. The system was used to recommend commodity products such as music and books. The advantage of using this method is that the buyer can get the best product as recommended by people who have similar buying profile.

### **4.2 Better Business Bureau (BBB)**

This approach is in the Merchant Brokering stage of the CBB Model. It employs a distributed trust and reputation mechanism that works by rates given to each other from both parties (seller and buyer) after a transaction on how well the other party managed his or her half of the deal, such as accuracy of product condition, and completion of

transaction. Other agents then can use these ratings to determine if they should negotiate with agents whose owners fall below a certain specified reputation threshold.

### 5 Proposed Framework

The proposed framework focuses only in the Product Brokering and Merchant Brokering stages of the CBB model. The available approaches used are the Automated Collaborative Filtering (ACF), and Better Business Bureau (BBB) methods to rank the products and merchants respectively.

#### 5.1 Software Agent Framework

The two sets of rankings in ACF and BBB are combined to obtain what is called, the "Consideration Set". From this set, Price Requests are made to the various merchants along with other constraints, such as number of item needed, date of delivery, and highest price acceptable. After that, a "Filtered Consideration Set" is obtained when there are merchants that cannot meet the criteria for certain products in the set.

After obtaining the "Filtered Set", the merchant with the product of the lowest price in the set will be selected for the buyer to continue to close the deal. In addition, the best-ranked merchant with the best-ranked product - in a diagonal manner (top-left to bottom-right). If there are equal prices in the same precedence category, the best-ranked merchant will be considered the best candidate to continue to do business with. This is simply because it is better to do business with a trusted merchant.

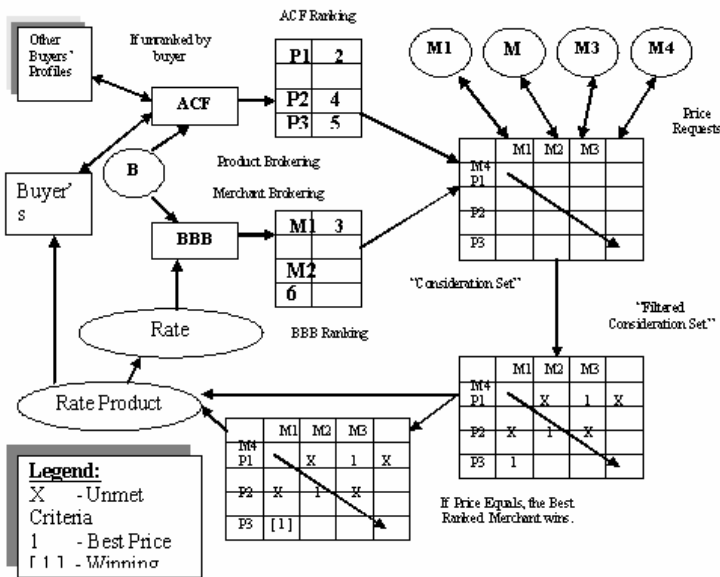


Fig. 1. The proposed combined approach for software agent buying decision process

Then, after the transaction has been completed, the buyer will need to rate the product or services that has been received so that it will be stored in the Buyer's Profile to be used as ACF reference in future decision-making. After that, the buyer also needs to rate the Merchant in terms of its trustworthiness, reliability, support and reputation to be used for the BBB ranking.

## 5.2 Discussion

There are some situations that should be highlighted by using this framework. One biggest drawback in the framework is that there might be situations where all instances in the "Consideration Set" might not meet the requested criteria. Such a situation is currently unresolved. Value Added Services (VAS) offered by the merchants such as bonus gifts, promotional items, and after-sales support are also not taken into account and will be overlooked by the user in the buying decision process. This can result in quite a big difference in the long run.

The framework does not cover the negotiation stage of the CBB model. There are still no software agents that have been developed that can effectively cover all of the stages in the CBB model. Most of them only focus in one or two stages of the CBB model. Another issue is that the main assumption for the best offer is the lowest price the merchant in the "Consideration Set" can offer following the precedence from best ranked to the lowest-ranked. It clearly overlooks other aspects in defining what is meant by best offer. Such aspects that might also be considered together with having the lowest price are delivery date and VAS.

## 6 Conclusions

This paper has described the traditional business model as well as the buying decision processes. The CBB model have been presented and considered. A framework that integrates ACF and BBB approaches are proposed for an effective buying decision process to be used by software agents. This framework will enable the agents to obtain the best price for a good product effectively. Furthermore, the framework will also assure that the transaction will be done with a reputable merchant.

## 7 Future Works

This project plans to implement the proposed framework in a large-scale multiagent electronic marketplace. Other situations will also be researched into such as non-cooperative agents in the marketplace, and the computational complexity of a simulated marketplace that employs the proposed framework.

## References

1. Guttman, R., Moukas, A., Maes, P.: Agents as Mediators in Electronic Commerce. *International Journal of Electronic Markets*, Vol. 8, No. 1 (1998)
2. Lomuscio, A. R., Wooldridge, M., Jennings, N. R.: A classification scheme for negotiation in electronic commerce. In: Dignum, F., Sierra, C. (eds.): *Agent-Mediated Electronic Commerce: A European Perspective*. Springer-Verlag (2000)

3. Nwana, H., Rosenschein, J., Sandholm, T., Sierra, C., Maes, P., Guttman, R.: Agent-Mediated Electronic Commerce: Issues, Challenges, and some Viewpoints. Proceedings of the Workshop on Agent Mediated Electronic Trading (AMET'98). Minnesota (1998)
4. Genesereth, M. R.: Software Agents. *Communications of ACM (CACM)*, Vol. 37, No. 7 (1994)
5. Franklin, S., Graesser, A.: Is it an Agent, or just a Program?: A Taxonomy for Autonomous Agents. Proceedings of the Third International Workshop on Agent Theories, Architectures, and Languages. Springer-Verlag: Berlin (1996)
6. Schneider, G. P., Perry, J. T.: *Electronic Commerce: Second Annual Edition*. Course Technology, Canada (2001)
7. Turban, E., Lee, J., King, D., Chung, H. M.: *Electronic Commerce: A Managerial Perspective (International Edition)* Prentice Hall, New Jersey (2000)
8. Blake, M. B.: Innovations in Software Agent-Based B2B Technologies. Proceedings of the Workshop on Agent-Based Approaches to B2B at the Fifth International Conference on Autonomous Agents (AGENTS 2001). Montreal (2001)
9. Bichler, M., Segev, A., Beam, C.: An Electronic Broker for Business-to-Business Electronic Commerce on the Internet. *International Journal of Cooperative Information System*, Vol. 7, No. 4 (1998)

# Improving Adaptability and Transparency of Dynamically Changing Mobile Agent Runtime Environments

JinHo Ahn<sup>1</sup> and SungMin Hur<sup>2</sup>

<sup>1</sup> Dept. of Computer Science, Kyonggi University  
San 94-6 Iuidong, Yeongtonggu, Suwon-si Gyeonggi-do 443-760, Republic of Korea  
Tel.:+82 31 249-9674

`jhahn@kyonggi.ac.kr`

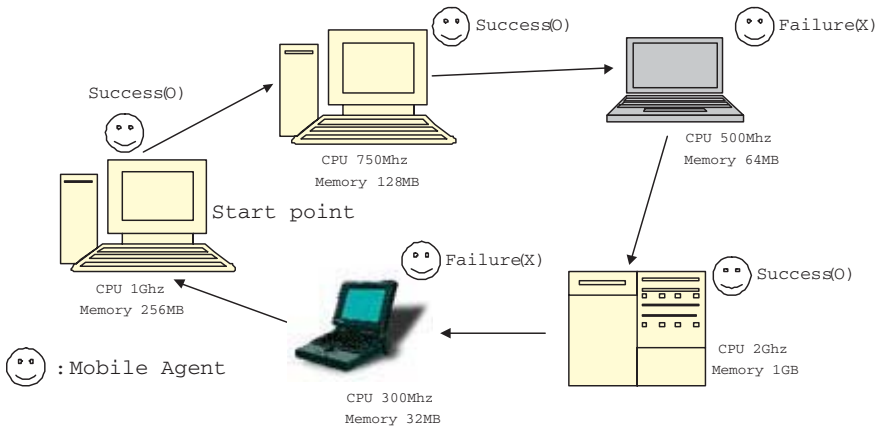
<sup>2</sup> Entrue Consulting Partners, LG CNS  
SFC Building, Taepeungro 1-84, Chunggu, Seoul, Republic of Korea  
`smhur@lgcns.com`

**Abstract.** The rapid emergence of small connected devices with wireless links needs highly dynamic adaptable distributed system architectures. But, in most current mobile agent systems, each mobile agent is able to exploit only uniform functionalities supported in every runtime environment. This feature has the agent difficult to use environment-specific resources. This paper presents a transparently dynamic adaptation framework using aspect oriented programming technique to adjust not only a variety of static resources, but also dynamic ones whose amount is continually changed at runtime even in the same computational environment. To make agent programmers easy to implement applications with no knowledge of dynamic adaptation, software developers in the proposed framework are classified into three groups, mobile agent application programmer, policy decision maker and component implementer. In here, policy decision makers can apply various adaptation policies to dynamically changing environments in order to accommodate mobile agents to the change of their resources.

**Keywords:** mobile agent, dynamic adaptation, transparency, aspect-oriented programming.

## 1 Introduction

The rapidly increasing availability of small connected devices with intermittent and low-bandwidth links enables end-users to take many opportunities to easily access, if necessary, tens of thousands of computing resources. On the other hand, this technological trend makes the distributed computing structures more complex, heterogeneous and dynamic [1,6,9]. In addition, mobility of users and devices leads to their softwares being executed on dynamically changing environments that support different capabilities and types of available local resources respectively. In terms of software development, these issues make it difficult to



**Fig. 1.** An example that the partial execution of a mobile agent on its movement route fails due to the heterogeneity of its computational environments

design the application programs because it is impossible to obtain completely accurate information about their dynamically changing runtime environments in advance. Therefore, a new middleware platform is required for enabling software components to adapt to their local runtime environments.

Thanks to their beneficial characteristics, i.e., dynamicity, asynchronicity and autonomy, the concepts of mobile agents is very attractive to satisfy the middleware requirement mentioned above [4]. Mobile agents are autonomous objects that are able to move from node to node in a computer network [4,7]. When an agent attempts to migrate to another node, the agent’s code, data and execution state, if needed, are captured and transferred to the next node, where the agent is resumed after arrival. However, this beneficial technology cannot become the practical alternative until it copes with the discrepancies among its changing environments. To solve the problem, many previous works primarily use resource abstractions or virtualizations, such as virtual machines or runtime systems for mobile agents [8]. If mobile agents use only common functionalities supported in all execution environments, they work well. But, as they cannot access uncommon functionalities specific only to a few environments, the environment-dependent resources may be unavailable to the agents. For example, assume that a mobile agent is written to be able to execute only on the nodes with at least 750 Mhz CPU and 128 MB main memory in figure 1. In this case, the agent fails to execute on the third and the fifth nodes in its whole movement route. Therefore, the mobile agent-enabled infrastructure should be designed to accommodate a variety of static resources like heterogeneous hardwares, operating systems or system configurations, but also dynamic resources like CPU, memory, network bandwidth whose amount is continually changed even in the same environment. The common way to solve the problem is static customization to use an if-then-else construct with conditional branches interchangeably executed depending on the computational environment [3]. But, this

solution is not very efficient under certain cases because the code with the big implementation may be rarely used while moving over the network.

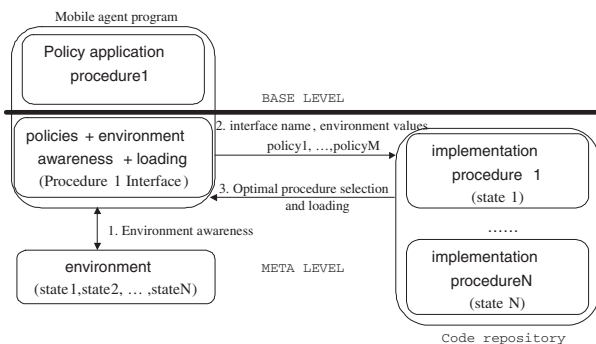
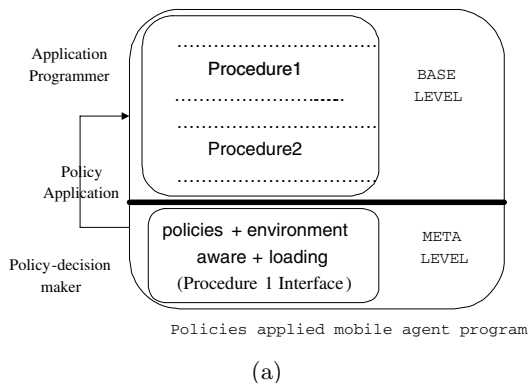
To address the drawback, Brandt et al. [2] introduced a dynamic framework to adapt a mobile agent to its currently running environment by using adaptors for identifying, loading and integrating environment specific parts into the mobile agent. The adapters include the context awareness module and the reconfiguration component. It improves efficiency of mobile code in terms of bandwidth through reducing the size of the movable code called the core part. However, the framework requires mobile agent application programmers to recognize dynamically changing parts. Moreover, it can apply only a single policy to the corresponding resource at runtime. So, if the state of each dynamic resource such as available memory size is changing at runtime and then its current implementation becomes non-executable in the state, the mobile agent may not continuously perform its task any longer because the implementation of the resource is determined only once when the mobile agent arrives at the current execution environment. This paper proposes a transparent adaptation framework using aspect oriented programming technique to enable policy decision makers to apply multiple policies to dynamically changing environments for adjusting mobile agents to the change of their resources. Unlike previous works, this framework divides roles of software developers into three groups to relieve application programmers from the complex and error-prone parts of implementing dynamic adaptation and allowing each developer to only concentrate on his own part. Second, this framework can accommodate not only various static resources, but also dynamic ones whose states are continually changed at runtime even in the same execution environment.

## 2 The New Framework for Transparently Dynamic Adaptation

In the existing dynamic adaptation framework [2], a mobile agent is divided into environment-dependent adaptable parts and an environment-neutral core part. The adaptable parts are exchanged in order to be suitable for the current computational environment. The core part is composed of functional code and non-functional code. In here, the functional code contains the application domain logic of the agent and the non-functional code generally includes capabilities of context awareness and reconfiguration. The core part moves from one node to another as a vehicle for the computational flow. So, the core part can be used as boot-strapper for the dynamic adaptation. However, this framework has two practical problems as follows: As the first problem, the scheme only considers the static resources whose states are initially set in a particular environment according to its single policy, but changed no longer. Second, it doesn't achieve complete transparency to application programmers because they must understand beforehand what parts of the mobile agent are environment specific.

To solve the problems, we present the following dynamic adaptation framework. First of all, agent programmers develop only the functional code of the core





**Fig. 2.** The proposed dynamic adaptation framework

part at the base level like in figure 2(a). Then, policy-decision makers recognize a part of the functional code that may be affected by underlying heterogeneity and needs dynamic adaptation, determine which execution policies can be applied for the part and then write the corresponding non-functional code, including the applicable policies, interface names, environment awareness module, dynamic implementation loading module and so on, at the meta level. In here, our framework uses aspect-oriented programming technique to achieve the seamless role separation because of the inherent conceptual separation it makes between the base level and the meta level [5]. If a monolithic mobile agent with the whole code for all different environments written in if-then-else style is intended to use for dynamic adaptation, the size of its migrated code becomes extremely larger and it is very difficult to transparently modify and add new adaptable parts to the agent program. The proposed dynamic adaptation procedure with multiple policies applied is explained in figure 2(b). When the core part of a mobile agent arrives at a new execution environment, it senses the environment and obtains environment-dependent variables from both the local repository and the

infrastructure monitor. Afterwards, the dynamic loading module at the meta level is performed with interface names, environments variables and applicable policies to determine the appropriate adaptable parts for the local environment, find and load the corresponding implementation procedures from the code repository. For this dynamic adaptation, component developers have to implement not only the corresponding interfaces, but also their associated condition functions, which are invoked with values of the environment variables related to the interfaces and return true or false according to whether the values are suitable for the interfaces. After checking and comparing all return values of the condition functions depending on their applicable policies, it is determined which implementation class is suitable for this environment. Then, the adaptable parts and the core part are assembled to form a mobile agent being able to execute its partial task on the new node.

### 3 Discussion and Concluding Remarks

Currently, we are implementing a simple prototype for realizing our framework using *uCode* [8], which is a lightweight and flexible toolkit for code mobility written in pure Java programming language. In addition, AspectJ [5], a simple and practical aspect-oriented extension to Java, are used to include the aspect-oriented programming technique into the framework. The prototype library consists of a Java package named *Adaptation*. This package is composed of three core classes and one interface as follows; *Place* class instantiates a mobile agent execution environment extending *uServer* in *uCode* toolkit. It also plays a role of a code repository maintaining classes for adaptable parts of mobile agents and delivering them to the dynamic loading module. For this purpose, each shared class space associated with a *uServer* is modified to implement the code repository. *Context* class consists of concerning environment variables, e.g., OS name, CPU type, version of Java Virtual Machine, network bandwidth, CPU utilization and so on, and functions for getting values of the variables. In here, each corresponding function is declared in the form of *get\_CONDITION(Context.VARIABLENAME)*. For example, if a variable is defined as *OS\_NAME*, function *get\_CONDITION(Context.OS\_NAME)* is invoked. Thus, when a new context is added, mobile agent programmers have only to use the corresponding variable in *Context* class and component developers just use the added function without implementing a new one. *AdaptationPolicy* interface defines the following constants representing various policies. The *AND* policy is suitable in case that all return values of condition functions for the environment variables applied are true. The *OR* policy is suitable in case that at least one among all return values of condition functions for the environment variables applied is true. The *NONE* policy is suitable in case that all return values of condition functions for the environment variables applied are false. Finally, *Adaptation* class implements kernel modules of our framework for applying various adaptation policies transparently, sensing execution environments and dynamically loading the corresponding implementation classes from the repository. In particular, function *setAdaptation()* is defined to

apply policies for a method of a mobile agent to its current runtime environment. If this function with a interface name, concerning environment variables and several policies is invoked, the adaptive implementation of the method is automatically linked to the mobile agent. Also, this class has a function enabling it to access to its local *Place* class.

However, this prototype is not a full-fledged mobile agent system yet. Also, for performance evaluation, we should implement some previous works, e.g., traditional monolithic adaptation framework where a mobile agent is migrated with the whole code for all heterogeneous execution environments. If all of them have been completed, we will attempt to compare our framework with the previous works with respect to the following two performance indexes: the first is the runtime overhead resulting from the execution of context awareness and the time for loading the implementation classes. The second is the size of migrated code of a mobile agent, which significantly affects network bandwidth.

## References

1. P. Bellavista, A. Corradi and C. Stefanelli. The Ubiquitous Provisioning of Internet Services to Portable Devices. *IEEE Pervasive Computing*, Vol. 1, No. 3, pp. 81-87, 2002.
2. R. Brandt and H. Reiser. Dynamic Adaptation of Mobile Agents in Heterogeneous Environments. *In Proc. of the International Conference on Mobile Agents*, LNCS 2240. pp. 70-87, 2001.
3. A. Duncan and U. Hölzle. Load-Time Adaptation: Efficient and Non-Intrusive Language Extension for Virtual Machines. *Technical Report TRCS99-09*, University of California at Santa Barbara, April 1999.
4. A. Fuggetta, G.P.Picco and G. Vigna. Understanding Code Mobility. *IEEE Transactions on Software Engineering*, Vol. 24, No. 5, pp. 342-361, 1998.
5. G. Kiczales, E. Hilsdale, J. Hugunin, M. Kersten, J. Palm and W. Griswold. An overview of AspectJ. *Lecture Notes In Computer Science*, Vol. 2072, pp. 327-353, 2001.
6. M. Parashar, H. Liu, Z. Li, V. Matossian, C. Schmidt, G. Zhang and S. Hariri. AutoMate: Enabling Autonomic Grid Applications, *Cluster Computing: The Journal of Networks, Software Tools, and Applications*, Special Issue on Autonomic Computing, Kluwer Academic Publishers, Vol. 9, No. 1, 2006.
7. V. Pham and A. Karmouch. Mobile Software Agents: An Overview. *IEEE Communications Magazine*, Vol. 36, pp. 26-37, 1998.
8. G. P. Picco.  $\mu$ Code: A Lightweight and Flexible Mobile Code Toolkit. *In Proc. of the 2nd Int. Workshop on Mobile Agents*, pp. 160-171, 1998.
9. D. Soldani, N. Lokuge and A. Kuurne. Service performance monitoring for EGPRS networks based on treatment classes, *In Proc. of the Twelfth IEEE International Workshop on Quality of Service*, pp. 121-128, June 2004.

# AgentAssembly: The Agent Framework Platform

Ockmer L. Oosthuizen<sup>1</sup> and E.M. Ehlers<sup>2</sup>

<sup>1</sup> Academy for Information Technology, University of Johannesburg, Auckland Park  
Kingsway Campus  
P.O. Box 524, Auckland Park, 2006  
South-Africa  
oo@adam.rau.ac.za

<sup>2</sup> Academy for Information Technology, University of Johannesburg, Auckland Park  
Kingsway Campus  
P.O. Box 524, Auckland Park, 2006  
South-Africa  
eme@rau.ac.za

**Abstract.** This paper discusses the AgentAssembly architecture and specifically the agent framework platform component of the architecture. An introduction is given to the work covered in the paper and previous work (see [1]) this paper is based upon. Then an in-depth discussion is rendered on the agent framework platform provided by the AgentAssembly architecture. The paper concludes with some remarks on the research presented and future aims of the ongoing research project.

**Keywords:** Agent programming languages, frameworks, and toolkits meta-modeling and meta reasoning.

## 1 Introduction

The Internet has evolved in recent years from a resource providing primarily information about certain topics to an active environment delivering raw data, information and services. The volume of information and services available is staggering and, sometimes, overwhelming.

This abundance of resources available on the Internet and its dynamic, ever changing nature has made it a prime target environment for the development of intelligent agents and multi-agent systems. Past research by various authors [2,3] has mainly focused on building custom agent-based systems to solve specific problems. There is, however, a need for a generic architecture for agents that use the internet as a functional domain, called *TeleAgents*<sup>1</sup> in our work.

The goal of the research undertaken is to apply component based design of software systems that has long been recognized in the realm of software engineering to agent-based systems<sup>2</sup>. In previous work [1], we proposed the AgentAssembly

---

<sup>1</sup> See our previous work [1] for a discussion on TeleAgents.

<sup>2</sup> See Brazier, et al. [4] for a discussion of the principles and benefits of component based design of Intelligent Agents.

framework to: 1) help facilitate the modular design and construction of agent based systems through the use of components and 2) to provide a standardised interface to the framework and agents constructed with the framework. This is achieved in the proposed framework by the Agent Framework Platform and the Agent Programming Interface (AGPI) respectively.

In this paper, the main focus of discussion will be the Agent Framework Platform. In section 2, we give a brief overview of the AgentAssembly architecture introduced previously. In section 3, the agent framework platform will be discussed in detail. The agent framework platform has the responsibility of providing core services and structures to agent designers and implementers. These core services include access to a library of standardised components, a set of default architectures and profiles for agents, a component integration mechanism, agent role configuration and a standard interface to the framework. Finally this paper concludes with some initial findings on the work presented and also discusses further work that is to be undertaken in the realisation of the AgentAssembly architecture.

## 2 AgentAssembly System Overview

The AgentAssembly can be described from two perspectives, the architecture perspective and the agent perspective. The agent perspective deals with the individual agents produced with AgentAssembly and the architecture perspective deals with the actual production process, services and components utilized. The three layers in the AgentAssembly architecture are (1) *Components/Components specializations and Agent Default Profiles* that lies at the heart of the AgentAssembly architecture and can be classified as either elemental process/knowledge components or default agent profiles. (2) *The Agent Framework Platform* leverages on the components/profiles and provides managed access to these components, an integration facility between components and between component groups and agent profiles, an agent role configuration and scripting facility and a standardized construction process. (3) *The Agent Programming Interface (AGPI)* defines a set of commands and methods to control and manipulate the framework platform described above and the agents created with the framework.

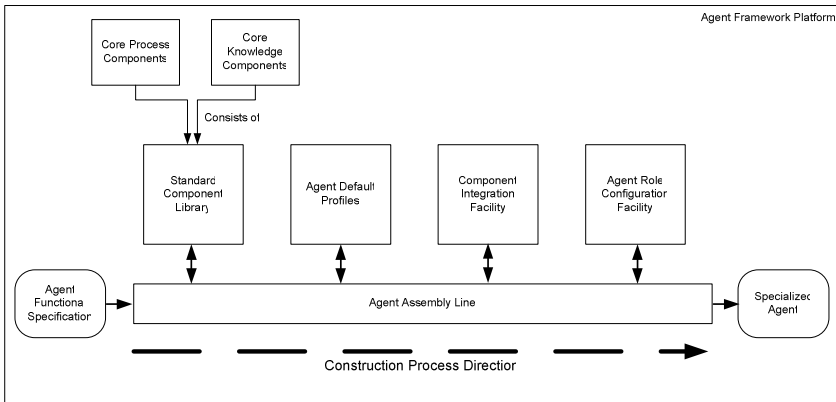
The agent perspective focuses on an individual agent produced by the framework platform. If the agent is to be used successfully in its target domain, certain aspects of the agent must be customizable and configurable. Through the same standard interface provided by the AGPI, the agent exposes methods for customization of agent attributes, process sequencing scheduling and agent control/execution monitoring.

The end product produced by the AgentAssembly platform is a fully functional individual agent. The agent's behaviour and configuration is specified by the role configuration activity in the construction phase, and an optional AGPI script that specifies any additional behaviour. Additional dynamic scripting can also be done through a standard AGPI interface to the agent.

The main benefit of such a standardised interface to the produced agents is that the agents can be controlled and monitored in the target environment with relative ease and flexibility.

### 3 AgentAssembly: Agent Framework Platform

The agent framework platform provides agent designers and implementers with the basic building blocks for building agents specifically for the Internet as well as a standard process of construction. There are five primary development goals behind the agent framework platform [1]: The first goal is *access to a library of standard components*. The framework should provide access to a library of standard components that can be reused in different agent projects. The second goal is *access to a set of default agent architectures or profiles*. The framework should provide a set of generic agent architectures to assist in organising components into a defined structure. The third goal is *a component integration mechanism*. The framework should then provide a method of associating components with each other and with the selected architecture. Knowledge components must be integrated with process components and the sequencing of processes to achieved desired results must be defined. The fourth goal is an *agent role configuration capability*. The framework should provide some type of mechanism to enable the configuration of the variable aspects of both the components used and the general architecture selected. Finally the framework should provide *an assembly line to help facilitate construction*. The framework should also provide a standard interface to assist and guide the implementer in the construction process. The services listed above form the basis of the proposed agent framework platform design given in Figure 1 below.



**Fig. 1.** Agent Framework Platform Elements and Construction Process [1]

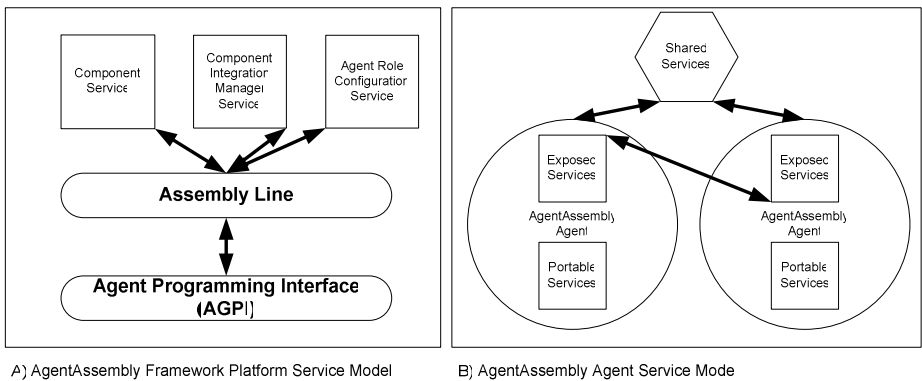
Figure 1 illustrates the agent construction process as a series of 5 Steps. Step 1 initiates the process with a functional specification of the agent’s desired capabilities and architecture. Step 2 involves interfacing with the assembly line. Various core process components and knowledge components are selected to achieve the desired functionality. Step 3 includes the selection of an appropriate architecture for the agent from a set of default architectures or alternatively a definition of a new custom architecture. Step 4 involves component integration between components and the selected architecture. Finally, step 5 involves pre-deployment configuration of various process

variables and architecture variables to equip the agent to function adequately in an environment such as the Internet.

### 3.1 Framework Platform System Organization

The AgentAssembly framework platform can be described as a service provider for building agents. The services provided by the platform correlates precisely with the five framework platform elements given in Figure 2. In their paper, Papazoglou and Georgakopoulos define services as self-describing, open components that support rapid, low-cost composition of distributed applications [5].

From an organization perspective, the AgentAssembly is service-orientated. Service-oriented computing (SOC) is defined as the computing paradigm that utilizes services as fundamental elements for developing applications [5]. The services available in the overall AgentAssembly architecture consist of four categories: basic construction services (Framework Platform), portable services, exposed services and shared services. These services and their contexts are illustrated in Figure 2 (a) and 2(b) below.



**Fig. 2.** AgentAssembly System Organization

The above figure depicts the role of services in the AgentAssembly system from both a construction perspective (a) and an agent perspective (b). In Figure 3(a), the developer interfaces through the use of AGPI commands with the agent assembly line, which acts as a request broker and service manager. The assembly line interprets requests and acquires resources from the basic construction services shown for the requests. These services include a component access service (components and agent profiles), an integration service and finally a role configuration service. The services functions correlate with the descriptions given earlier.

From the agent perspective, agents leverage on shared services that supply knowledge/processing capability to individual agents. Portable services constitute the plug-gable components selected in the construction phase. Perhaps more interesting, are the exposed services published by each agent. Exposed services make it possible to develop multi-agent systems (MAS) with the AgentAssembly platform and form the

basis for inter-agent communication and co-operation. Agents can also share knowledge and capabilities through merely publishing them as exposed services.

With the discussion of the AgentAssembly system organization given above and the system components covered in the previous sections, the components and services that form the core of the AgentAssembly system can now be discussed in the next subsection.

### 3.2 Framework Platform System Components

The components of the framework platform system are the agent functional specification, component/agent default profiles library service, the component integration service, the agent role configuration service and finally the agent assembly line.

The agent construction process starts with an agent *functional specification*. The functional specification indicates to the framework platform the characteristics and abilities that the agent under production should possess. The functional specification is then submitted to the component/agent profile service through the assembly line where the appropriate components are selected according to the specification. The *component/agent profiles service* enable component and agent profile selections based on the information contained in the functional specification discussed in the previous section. This implies that the service should be “intelligent” enough to match functional requirements with components stored in the component/profile store. This “intelligence” could be achieved in one of two ways. Firstly, the individual components could be stored in the component store with an accompanying metadata sheet encoded in XML. A keyword-driven text query, based on the functional specification could then match potential components/profiles. A disadvantage to this approach is that a keyword based search is often lacking in precision and expressiveness. A second approach would be to define a standard reference set of capabilities and profiles. These descriptions are then encapsulated inside components/profiles and components can merely be polled to establish its suitability. The advantage of this is that components can be precisely matched with requirements. After component selections the *integration service* integrates components with each other and the selected profile. If a mark up approach is taken to the description of components, the interface expected by each individual component could then also be described. This would then enable the integration service to automatically match up components with other components as long as the interface specification is met in terms of exposed methods and data type. It is important to note however that complete automation of the integration phase is not ideal. The integration system should act more as a recommender system, recommending a suggested integration but allowing the developer freedom to manually configure any components as required. The *agent role configuration service* represents the last phase before deployment of the newly constructed agent is its role configuration. Here, the service interfaces with the agent and compiles a list of configurable properties through polling of its internals. This list is then matched with values or settings in the functional specification or manually defined by the developer. Perhaps the most important task of the configuration service is the final classification of the various service types available in the produced agent. Here, services are classified as shared, exposed or portable as discussed previously. Finally, the *agent assembly line* is used throughout the construction phase and acts as a request broker and mediator between



services and the developer. The assembly line contacts the service and exchanges control information to drive the agent construction process. The assembly line could be seen as being an agent in its own right, intelligently and, nearly, autonomously guiding the construction process.

## 4 Conclusion and Further Research

In this paper the AgentAssembly system was discussed and, more specifically the framework platform component of the AgentAssembly system. In section 2 a brief background of previous research and goals behind AgentAssembly was given. Section 3 focused on the Agent Framework Platform and a discussion of the service-orientated system organization model was given. Additionally the five key components of the framework platform system namely the agent functional specification, component/agent default profiles library service, the component integration service, the agent role configuration service and finally the agent assembly line was also discussed.

Future work will focus on a realization of the framework platform as described in this section through the use of web-services. This would effectively enable the service orientated architecture presented in this paper. Other work will focus on technical design of the components/agent profiles and also a comprehensive description and specification for the syntax and usage of the Agent Programming Interface (AGPI).

## References

- [1] O.L. Oosthuizen, E.M. Ehlers, Towards a Component-Based Architecture for TeleAgents, In *Proceedings of the 8<sup>th</sup> Pacific Rim Conference on Multi-Agents (Prima2005)*, Kuala Lumpur, Malaysia, 2005.
- [2] N. Kravtsova and A. Meyer, Searching for Music with Agents, *Lecture Notes in Computer Science*, Volume 1931, 2000.
- [3] O. Etozioni and D. Weld, A Softbot-Based Interface to the Internet, *Communications of the ACM*, Vol.37, No. 7, July 1994.
- [4] F.M.T. Brazier, C.M. Jonker and J. Treur, Principles of component-based design of Intelligent Agents, *Data and Knowledge Engineering*, 41(1): 1 – 27, 2002.
- [5] M.P. Papazoglou and D. Georgakopoulos, Service-Oriented Computing, *Communications of the ACM*, Vol. 46, No. 10, October 2003.

# A Multi-agent Architecture for CSCW Systems: From Organizational Semiotics Perspective

Wenge Rong and Kecheng Liu

Informatics Research Centre, University of Reading,  
Whiteknights, Reading, RG6 6AH, UK  
{w.rong, k.liu}@reading.ac.uk

**Abstract.** This paper describes a multi-agent architecture to support CSCW systems modelling. Since CSCW involves different organizations, it can be seen as a social model. From this point of view, we investigate the possibility of modelling CSCW by agent technology, and then based on organizational semiotics method a multi-agent architecture is proposed via using EDA agent model. We explain the components of this multi-agent architecture and design process. It is argued that this approach provides a new perspective for modelling CSCW systems.

**Keywords:** CSCW, Organizational Semiotics, EDA Agent, Architecture.

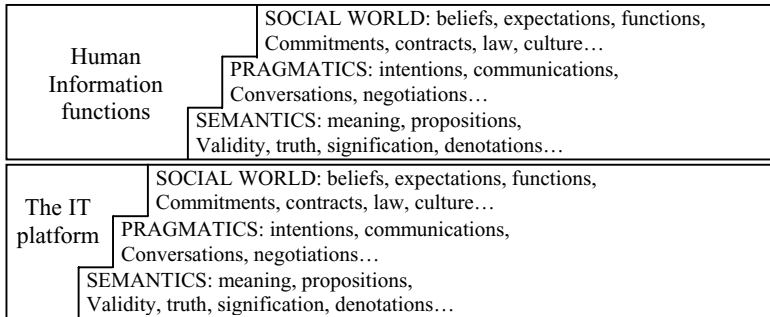
## 1 Introduction

Teamwork has been attached more and more importance to make organizations successful. An organization process normally involves different people from different location, which makes teamwork more difficult. Fortunately, the increased availability of computer technologies and advances in artificial intelligence make it possible for people to work together. Around twenty years ago, computer-supported cooperative work (CSCW) emerged to support distributed group working integration. CSCW naturally involves a lot of human activities in more than one organization. That means it can be considered as a social system, in which members will be automatic to work independently and cooperate socially to achieve organizational goals. [7] argues that the most emphasized abilities of an agent or agent-based system should include autonomy, reactivity and social abilities. From this point of view, CSCW systems can be modelled as agent-based systems to solve the complex process-modelling requirement where only incomplete knowledge of the domain is available [8].

Since CSCW systems are very complex, dynamic and non-linear due to the high flexibility of human activity, it is difficult to be modelled properly. Organizational semiotics is a leading candidate in analysing organizations, based on this theory, a multi-agent architecture is proposed in this paper. This paper is organized as follows. Background information about CSCW systems, organizational semiotics theory and agent technology will be reviewed in section 2. Section 3 will investigate EDA agent. Section 4 will give detail design of the multi-agent architecture based on EDA agent. Section 5 illustrates design process and section 6 concludes the paper.

## 2 Background

The term CSCW was invented by Greif and Cashman in 1984 [4], but different scholars express CSCW systems in different ways. In this paper, we will accept Wilson's definition of CSCW as "a generic term, which combines the understanding of the way people work in groups with the enabling technologies of computer networking, and associated hardware, software, services and techniques" [10].



**Fig. 1.** Semiotics framework

Though various types of organizational models are proposed, most of them are not applicable to be used for modelling complex human organizations. In this paper, we propose a different approach based on organizational semiotics theory [15], which views an organization through six-level structure to study information sign, as depicted in figure 1. From this framework, we can see the separation of human activities and technique platform. The upper three layers mainly concerned with the use of signs, how signs function in communication and intentions, and what the social consequences are of the use of signs. Organizational semiotics is particularly useful in studying social norm and commitments, which are essential parts of CSCW systems.

The term agent usually refers to an entity which functions continuously and autonomously in an environment in which other processes take place and other agents exist, or an intentional system [13]. There are three main agent architecture, i.e. reactive agents [11], deliberative agents [7] and hybrid agents [5] in the community and a lot of agent theories such as BDI theory [1], ICE theory [2], Logic approach [6]. Intention theory [9], Knowledge and action theory [12] are proposed to study the intelligent aspect of an agent. However, traditionally agents are only considered as intelligent software components. To address the complex human behaviour, in this paper, we consider human beings in social organizations as human agents, which should be capable of interacting socially with other agents, including human users and other agents. One of the main features of CSCW is the sociability. That means within CSCW fields there maybe need specific agent theory and architecture.

### 3 EDA Agent

This model is proposed by [3], which uses semiotics approach to devise an agent framework, where normative knowledge and norm-based coordination is emphasized. Its main model components include epistemic, deontic and axiological components respectively. This norm-based structure reflects a social classification of norms and provides a principle base for agent structure. This model argues that this social model can be used both to analyse an organization and to guide the design of mixed organizations where human and artificial agents cooperate.

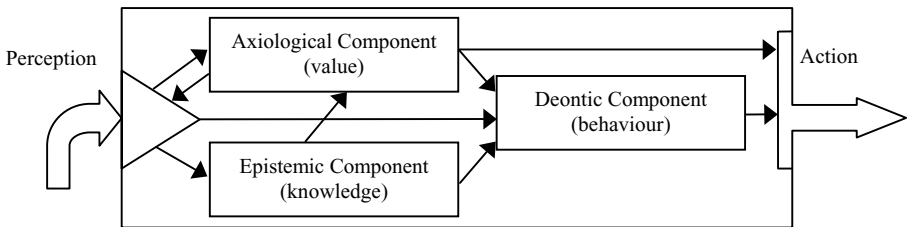


Fig. 2. EDA model component

### 4 General Multi-agent Architecture

Based on EDA agent and organization semiotics theory, a general multi-agent architecture for CSCW systems is proposed here, as shown in figure 3, which includes four types of agents, a data persistent layer, backend database and a meta-norm repository.

- 1) **Local agent:** Local agent represents the human agent in an organization. This type of agent is used to execute an action in a CSCW system and cooperate with other local agents through interactive agent. Furthermore, it can communicate with super agent and ask assistance to update its norm, or business rule. In our approach, the local agent has three level intentions. On the first level, the agent does not react to an external input unconditionally. It is able to select a certain action in each circumstance. On the second level, the agent should update its own norm gradually with the assistance of super agent, which is depicted below. The third level intelligence is the social level. Agent is not working alone. It works with other agents via coordination, cooperation, negotiation, etc. After learning other agents' concern via cooperation, the local agent can ask super agent to update its norm as well. Through super agent and meta-norm, the local agent holds evolving ability.
- 2) **Super agent:** Super agent is a control agent that is used to administrate local agent's action and to assist in achieving local agent's intelligence. When system is initially modelled, some meta-norms are defined. Meta-norms are rules to update local agents' norms. When a local agent executes certain actions, the super agent will monitor the success of the action and give feedback to local agent. Local agent can use these feedbacks to update its norm, namely the second level intelligence mentioned before. At the same time, the super agent will monitor the

cooperation of local agents and use the meta-norm to update certain local agent's internal norm.

- 3) **Interactive agent:** Interactive agent is used to assist the connection of different local agent and then help participants in their cooperative work. All local agents will register its ability through interactive agent. When a local agent asks cooperation, the interactive agent will try to find proper agents and create connection between these agents. Interactive agent is adopted because it makes the system simple and clean by simplifying the information flow.
- 4) **Meta norm repository:** Meta-norms are a special type of norms. Their format is same as normal norms but super agent use them to guide local agent's internal norm update.
- 5) **Data persistent layer and database:** This backend component provides data access service for local agents.
- 6) **Cooperation agent:** The figure 3 only depicts architecture within one organisation. Actually, it is easily extended to support multi-organization, e.g. CSCW systems. In CSCW systems, a cooperation agent monitors every different organization. When a local agent in one organization asks negotiation and no one agent within the organization is able to do so, the interactive agent will ask cooperation agent to broad request to other interactive agents. Once some agents are located, the local agent can negotiate with those agents.

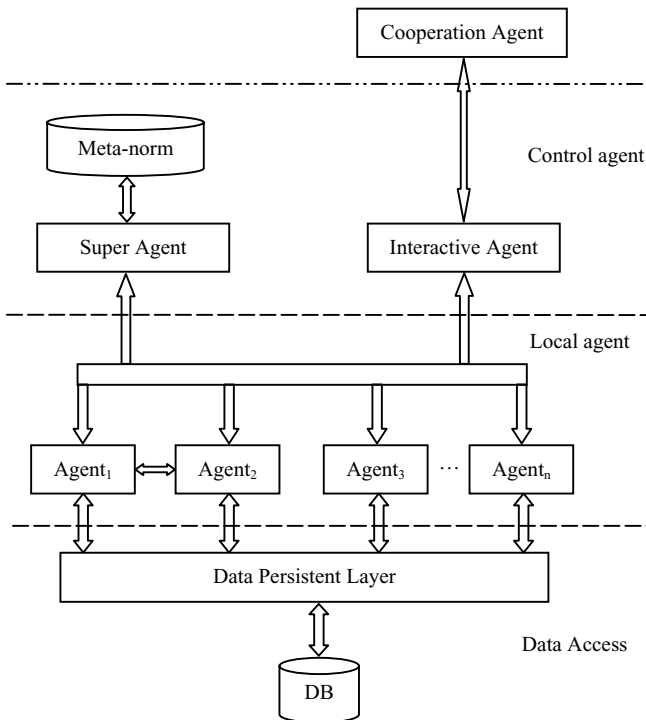


Fig. 3. General multi-agent architecture

## 5 Design Process

In this section, the overall design process will be depicted. The core steps in designing the multi-agent architecture are semantic analysis and norm analysis, which produce agents and norms and meta-norms, respectively.

The first and most important step is to candidate local agent. Organizational Semiotics is a framework to bridge the gap between organisations and computer based systems [14]. It provides a serial of conceptual tools to analyse the organizations. Semantic analysis is a fundamental component in organizational semiotics. It devises a canonical formalism with focus on the responsible agent and his repertoire of behaviour [15]. Figure 4 is a sample of ontology chart of an organization.

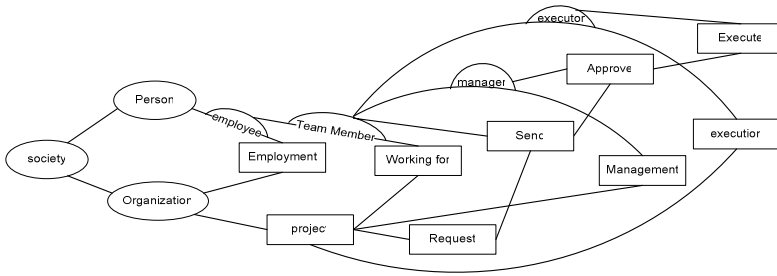
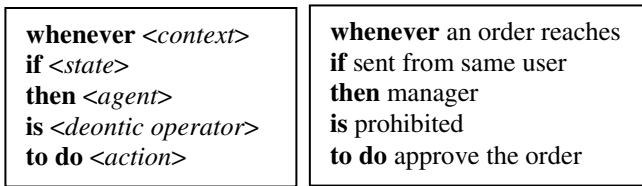


Fig. 4. Ontology chart



a. Norm format

b. Example

Fig. 5. Norm definition

After finding the agents, next essential step is norm analysis. Since the ontology chart defines the meaning of the concepts within organizations, there is a need of some mechanisms to describe how the agents in the ontology chart take appropriate actions and behaviours. Norm analysis can be used to capture each agent’s complex logic constraint and human intervention. Apart from defining norm related to each agent, norm analysis is also used to define meta-norms, which play vital role in achieving adaptability and extensibility of the whole CSCW systems.

The ontology chart depicts what pattern of behaviour is ontologically available, the detailed conditions and constraints of the realizations of the behaviour patterns are covered by norms. Norm is based on deontic logic and can be simply described as a

statement in a form as shown in figure 5(a) [15], [16]. There are three deontic operators “obliged”, “permitted” and “prohibited” defined to prescribe people must, may and must not do behaviours.

The norms are used to understand the patterns of the agent behaviour. Considering a purchasing workflow in a company, one requirement is that a person who is a manager cannot approve a request sent by him/her. Then, the norm can be defined as shown in figure 5(b). More details can be found in [15].

## 6 Conclusions

In this paper, a multi-agent architecture is proposed to support CSCW system modelling. This approach is from organizational semiotics perspective. The benefit is that it considers the CSCW systems as social model and takes into account of human agents. The detail structure and design process is elaborated and it is argued that this approach provide a new perspective on CSCW system modelling.

## References

1. Rao, A., Georgeff, M.: Modeling rational agents within a BDI-architecture. In Proceedings of Knowledge Representation and Reasoning, (1991) 473–484
2. Werner, E.: Logical Foundations of Distributed Artificial Intelligence, In O’Hare, G., Jennings, N. (eds.). Foundations of Distributed Artificial Intelligence, Wiley-Interscience, New York, (1996)
3. Filipe, J., Liu, K.: The EDA Model: An Organizational Semiotics Perspective To Norm-Based Agent Design. In Proceedings of the Agents’2000 Workshop on Norms and Institutions in Multi-Agent Systems. Barcelona, Spain, (2000)
4. Grudin, J.: Computer-supported cooperative work: Its history and participation, IEEE Computer, vol. 27, no. 5, (1994) 19-26
5. Georgeff, M., Lansky, A.: Reactive reasoning and planning, In Proceedings of the 6<sup>th</sup> national conference on artificial intelligence, Seattle, WA, USA, (1987) 677-682
6. Singh, M., Asher, N.: Towards a formal theory of intentions. In Proceedings of the European Workshop JELIA, (1991) 472–486
7. Wooldridge, M., Jennings, N.: Agent theories, architectures, and languages: A survey In Intelligent Agents, In Proceedings of the ECAI-94 Workshop on Agent Theories, Architectures and Languages, (1995) 1-39
8. Jennings, N., Wooldridge, M.: Applications of agent technology, In Jennings, N., Wooldridge M.(eds). Agent technology: foundations, applications and markets, Springer-Verlag, (1998)
9. Cohen, P., Levesque, H.: Intention is choice with commitment, Artificial Intelligence vol. 42, (1990) 213-261
10. Wilson, P.: Computer supported cooperative work: an introduction, Intellect books, Oxford (1991)
11. Brooks, R.: Intelligence without reason, In Proceedings of 12<sup>th</sup> International Joint Conference on Artificial Intelligence, (1991) 569-595
12. Moore, R.: A formal theory of knowledge and action, In Hobbs, J., Moore R.(eds.). Formal Theories of the Commonsense World, Ablex Publishing Corporation, (1985)

13. Shoham, Y.: An overview of agent-oriented programming, In Bradshaw J.(eds.). Software agents, AAAI press, California, (1997) 271-290
14. Filipe, J., Sharp, B., Liu, K.: An intelligent Agent-Based Methodology for Legacy Information Systems Integration, In Proceedings of UKAIS, (1999) 297-306
15. Liu, K.: Semiotics Information Systems Engineering, Cambridge University Press, (2000)
16. Stamper, R., Liu, K., Hafkamp, M., Ades, Y.: Understanding the Roles of Signs and Norms in Organizations – A Semiotic Approach to Information Design, Journal of Behaviour and Information Technology, vol. 19 no.1, (2000) 15-27



# Knowledge Description Model for MAS Utilizing Distributed Ontology Repositories

Kyengwhan Jee and Jung-Jin Yang

School of Computer Science and Information Engineering  
The Catholic University of Korea, Yeouido Post Office, P.O. Box 960,  
35-1 Yeouido-dong, Yeongdeungpo-gu, Seoul, Korea (150-010)  
Tel.: +82-2-2164-4377; Fax: +82-2-2164-4777  
{sshine106, jungjin}@catholic.ac.kr

**Abstract.** The rapid growth of IT technologies enables the quality of human's daily life to be improved dramatically. Contrasting to the services in previous computing environment directed by user's request, the services in ubiquitous computing era of new IT technology are provided through recognizing user's intention and utilizing context-aware information suited to the user. According to the contextual information, agents need to set a dynamic goal to achieve and work collaboratively with other agents. Agents that take control over their behaviors with capability of communicating with other agents become a thrust in this up-coming computing environment. This work focuses on building ontologies, shared knowledge bases among agents, to improve semantical interoperability among agents. More attention is given to the construction and effective management of ontology repository along with its requirement and organization. Ontology agent suggested takes an initiative role to manage the repository in a distributed manner and to facilitate the use of ontology in a multi-agent environment.

## 1 Introduction

As different informational devices and applications sink into our life, the ubiquitous computing paradigm postulated by Mark Weiser is being realized one by one. For autonomous entities such as agents to interact with one another, they need to know, beforehand, what kinds of interfaces they need to know, what kinds of interfaces they support and what protocols or commands they understand. In a truly distributed scenario, such as the ubiquitous computing environment, it may not be reasonable to assume that such agreement exists. Although researchers of the ubiquitous computing share the presupposition that smart entities' autonomous and adaptive handling of the change of their surroundings improves the user's life, an agent, which is the primary entity that constitutes the Ubiquitous Computing, is not adaptively correspond to the distributed and dynamic environment as much as its cognitive ability of the environment to which it belongs. In this paper, the Knowledge Description Model, which is designed to increase the semantic interaction between various sources of knowledge and agents utilizing ontology, which will be the central data structure in the intelligent activities within the multi agent system, and the ontology repository which effectively administers the ontology, and the framework architecture that

supports the smooth utilization of the knowledge sources through the distributed ontology repository are presented. In particular, this paper focuses on providing a base structure for agents to retrieve knowledge from the ontology repository and produce new knowledge through the standard ACL (Agent Communication Language) technology. This enables agents, which communicate on the message basis, to accommodate the already described knowledge to cope up with changes.

## 2 Related Work

CONON [1], utilizing OWL, constructed an inferable context ontology and presented a generalized meta-ontology and an expansible, domain-specific ontology. CoBra [2], also utilizing OWL, proposed an architecture that is capable of inferring and sharing the context knowledge using a broker ontology on the basis of a lucidly described context ontology. By describing the information recognized by an agent utilizing ontology, the sharing and reutilization of knowledge were secured and the checking of the validity of knowledge models and the expansion of knowledge have been facilitated.

The ontology repository has already been recognized as the primary subject of research, and high capacity ontology repositories like KAON[3] and HAWK[4] provide the functions of modification, editing, and creation as well as storage/retrieval of data. An ontology repository should assist storing, creating, editing, searching, integration, version management, and consistency maintenance of data regionally constituting an ontology and should be capable of effectively storing/retrieving meta-information for inference and assisting the combination and synthesis of ontologies. Currently, the standard language describing ontology is OWL and most ontology repositories parse and store ontologies described in OWL/RDF in RDBMS. It uses diverse mechanisms to store the outcome of parsing in the database and provides various APIs for retrieving and searching the stored data. However, there is no methodology that can utilize the ontology repository at the agent level.

This research, extending the agent platform in the preceding research [5] to assign an ontology agent in charge of query to the ontology repository, renders the ontology agent and the agent platform independent of a certain application of platform by allowing the agent platform and the ontology agent to use the standard web service for the communication in the distributed environment in order to make query by converting ACL message from the agent and deliver it to the ontology repository. Also, this research presents the Knowledge Description Model (KDM) which corresponds to the ontology repository that allows the re-utilization of both assertive and procedural knowledge by ontologizing them on the basis of the Web Service which facilitated change and expansion. The Web Service is put to use as a means of communication in order to accommodate ontology repositories of different types.

## 3 Knowledge Description Model

An agent recognizes sets of information of multifarious sources according to the task it is to perform. The Knowledge Description Model is a model proposed to describe

the sets of information recognized by the agent. As the entities and knowledge base constituting the ubiquitous computing environment rapidly change with their various goals and policies, the KDM should have a structure capable of accommodating various kinds of knowledge.

### 3.1 Knowledge Description Model

The Description Instance illustrated in the left center of Fig.1 describes the set of information necessary for an agent to perform its task. The Description Class has *producedBy* and *consumedBy* relations with the Entity Class which produces and consumes the Description Instance, and the class corresponding to the range of each relation vouchsafes its consistency through the meta-ontology. An ontology describing the Entity Class may have diverse structures suited to the aim of its design and perform logical inference within its range for the guarantee of consistency and the expansion of knowledge. (i.e. the Behavior property in Fig.1 is described and guaranteed its consistency in the Entity Ontology.

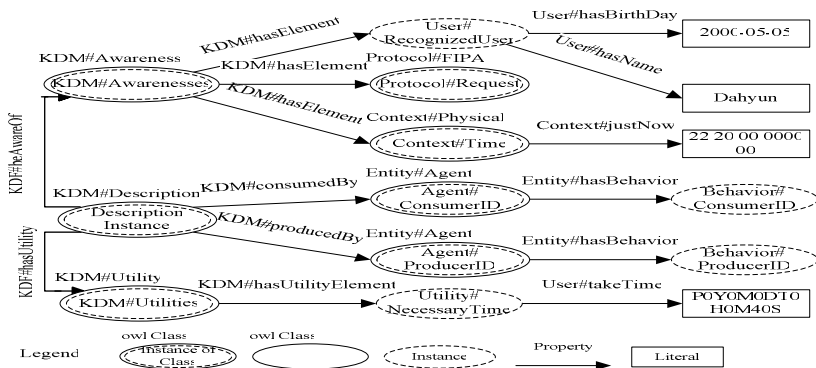


Fig. 1. Knowledge Description Model: Base URI is assumed as <http://idis.catholic.ac.kr/term/>

The Awareness class illustrated in the upper-left of Fig.1 has a *beAwareOf* relation with the Description class and is the subject of the assertions required for the entity consuming the Description Instance to perform computation. It should be possible for the entity consuming the Description Instance to receive the knowledge necessary for the computation from disparate ontologies and entities in the form of assertions. For this reason, the *hasElement* relation has the Awareness class as its domain and its range is not described.

The Utility class described in the lower-left of Fig. 1 has a *hasUtilityElement* relation with the Description class. The *hasUtilityElement* relation has the Utility class as its domain, and the range is not described so as to receive utilities suited to the feature of the Description Instance from diverse ontologies and entities in the form of assertions.

RDF triple assertions are described by the resources of the *Awareness* and *Utilities* instance. Thus, our approach can accommodate both cases where the knowledge is

expressed as subclass and instances. The knowledge re-expressed in : Class makes easy expansion, elucidated decision making, and various scenario modeling utilizing the knowledge detailed in subclasses possible.

The knowledge re-expressed in instance, by accommodating a vast amount of terms as instances, can obtain common understanding without much effort, although it has limitations in the expansion and range of utilization [6].

### 3.2 Scenario for Smart Media

In this section, both a Smart Media scenario and a scenario adapting to a new environment in the Smart Home environment are presented on the basis of the KDM.

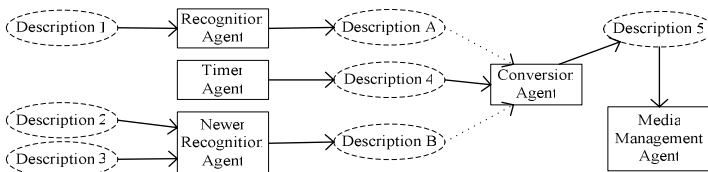


Fig. 2. Scenario for Smart Media Management Agent

**[Scenario 1: Smart Media].** The agent supervising the media children come across (e.g., Computer, TV, Audio) recognizes *Description 5* (The same as the *Description* in Fig. 1) illustrated in Fig.2. On the basis of the information of the user recognized (i.e. birthday, name) and situation (i.e. time), the *Media Management Agent* administers the multimedia and carries out appropriate instructions to the children according to time. In order to create *Description 5* consumed by the *Media Management Agent*, the *Recognition Agent* above all consumes *Description 1* to produce *Description A*, and the *Timer Agent* produces *Description 4* to deliver it to the consumer of each *Description*. And then, the *Conversion Agent* produces *Description 5* by consuming *Description A* and *4* and delivers it to the corresponding consumer.

**[Scenario 2: Entering New Entity].** The *Newer Recognition Agent* which consumes *Description 2* and *3* described in the new ontology participates in the Smart Home environment as a new entity. The *Newer Recognition Agent* produces *Description B* which is the same instance as *Description A* (The assertion of the resources having *Utilities* and *Awareness* as subject can differ). The utility of *Description A* and *B* is expressed in recognition rate and the *Conversion Agent* consumes *Description B* and *4* to produce *Description 5* and deliver it to the *Media Management Agent* for the *Conversion Agent* chooses a *Description Instance* of a high recognition rate.

## 4 Framework for uT Based on Distributed Ontology Repositories

The ontologies illustrated in the lower-left of Fig.3 have structures suited to the universe of discourse and are stored in repositories present in the distributed

environment. The entities constituting the Ubiquitous environment is guaranteed its interoperability with other entities and retrieve the knowledge described in ontology through queries based on meaning for the expansion and adaptation of knowledge. Agents send and receive queries and the results contained in SOAP message through HTTP protocol for interaction between entities and ontology repositories illustrated in Fig. 3. The ontology repository provides a query engine as the Web Service and the entity that retrieves knowledge from the ontology uses the Web Service Client as library. This paper adopted the most universal transmit protocol, which, as a generic way, is approachable to various ontologies accommodating knowledge contents.

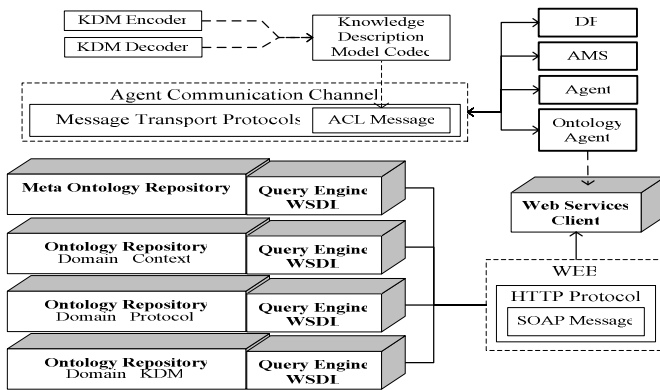


Fig. 3. Architecture for MAS based on Distributed Ontology Repositories

The entity that utilizes ontology in the Multi Agent System is the Ontology Agent (OA) and the OA provides ontology services for other agents, which use ACL messages. The OA is requested a *Knowledge Description* that is to be consumed by another agent and queries the ontology repositories to produce a corresponding *Knowledge Description*.

Agents interact with other agents using the ACL message. The *language* slot of the ACL message denotes KDM, the *ontology* slot of the ACL message denotes <http://idis.catholic.ac.kr/term/KDM#>, and the *content* slot of the ACL message denotes Knowledge Description. The Knowledge Description is stored in the *content* slot as RDF Triple set, and agents retrieve the values of subject, predicate, and object of triples through the KDM codec library.

We described the knowledge an agent knows in the KDM assuming that the agent is aware of the information recognized to perform actions appropriate to the purpose of its design in the viewpoint of itself through its own world model such as in BDI architecture. The Knowledge Description can be re-utilized through the Knowledge Description Ontology at the point of the agent design, and the re-utilization of agents using knowledge described in diverse ontologies can be induced.

## 5 Experiment

### 5.1 Knowledge Description Model Codec

Although it was possible to mark the boundary of sets when the resources possessing the *Awareness* as their subject were accommodated to describe the KDM, in this case, it required the limitation of the expression of the KDM ontology and additional interpretation of the List structure. Accordingly, we chose the approach which includes assertion in the method of declaring resources in the KDF Instance and altogether considers agents at the time of their design.

As a structure that is capable of storing, renewing, and retrieving the Triple set so as to input and retrieve the Description in ACL message, we declared the KDM root element, the Triple element which has no limitation of the number of its occurrence, and the child element of the Triple element as the triple of subject, predicate, and object respectively, in XML schema. On the basis of the declared schema, we converted the XML Schema into the form of Java class utilizing Java Architecture for XML Binding (JAXB) tool. Since there was no need to alter the tree structure of the XML Instance, we created the KDF Codec utilizing the JAXB Binding Compiler.

Agents unmarshal the Description expressed in the XML Instance contained in the *Content* slot of the ACL Message, which is to be consumed, into Java Instance, and handle the Triple set utilizing the *get* and *set* methods of the Instance. Agents perform marshaling in order to describe the Description it is to produce in the form of an XML Instance in ACL Message.

### 5.2 Simulation of Framework Based on Distributed Ontology Repositories

We used Minerva contained in the Semantics Toolkit (IBM) as the ontology repository and query engine, and transformed Minerva and the query engine into the Web Service using AXIS (Apache.org) and Tomcat (Apache.org). In order to make it possible for the ontology agent to inquire and retrieve the knowledge stored in Minerva through SOAP message, we used the AXIS Client library. Fig. 4 is the illustration of the observation of the query and response of the information of entities that produce the Knowledge Description using the SOAP Monitor.

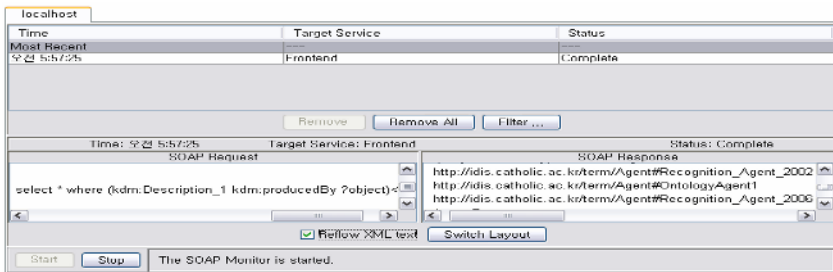


Fig. 4. Query the producer of the Description\_1 and query result

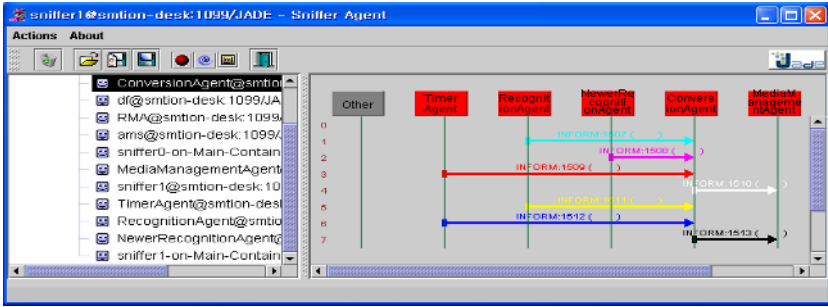


Fig. 5. Monitoring scenarios in section 3.2

The ontology agent produces the Description to be produced through the semantic query of the necessary information to deliver it to another agent, and provided a general method of approaching diverse ontologies through the Web Service.

The Sniffer Agent in Fig.5 shows a monitoring performed as for the agent system processes basing on the ACL message conveyed between agents. Fig 5 is scenarios in Section 3.2 which are embodied on the basis of JADE, and we omitted the embodiment of FIPA-Protocol on the purpose of simplifying the matter and focused on the interaction between agents using the KDM instead.

The *Timer Agent* in Fig. 5 conveys the *Description* (local time) to the *Conversion Agent* in fixed intervals, and the *Conversion Agent* selects the *Description* which has the highest utility among the *Descriptions* of the *Recognition Agent* and the *Newer Recognition Agent*; i.e. it selects the message of the line 3 between the messages of the line 2 and 3. in Fig. 5.

We chose the approach that does not render the function of description creation and validity confirmation ad hoc to the Architecture but commissions it to the entity constituting the Ubiquitous environment. In case of including assertions that do not require *Description*, we can keep an agent that optimizes *Description* in view or induce a description optimized by the utility of the *Description* through the competition between descriptions. The KDM, as a model describing domain-specific knowledge, can furnish a framework capable of containing information appropriate to the purpose of its design, increase the reutilization of the agent system, and guide agent organizations.

## 6 Conclusion

We have proposed a structure in which an agent queries and retrieves knowledge from the ontology repository through the Web Service and the Knowledge Description Model for smoothly accommodating the production and change of the knowledge to be used. In order to design an agent system on the base structure constructed likewise, we were able to reutilize the existing agent and the agent organization by inquiring an appropriate Knowledge Description. And we have presented a framework that flexibly includes new entities and knowledge repositories and performs adaptive

actions through the interaction with the existing entities. The Knowledge Description has the policy of including elements necessary for an agent to think without restraints, and consequently can change and be expanded flexibly. As a current work, we are at the stage of experimenting the communicational delays and synchronizations between agents that may be caused by accepting various policies, and are planning to extend our future research to the expansion of the agent knowledge as well as the reutilization of the agent structure by rendering the interaction protocol and solution etc. as knowledge and providing them in plug-in format.

## Acknowledgement

This research is supported by the ubiquitous Autonomic Computing and Network Project, the Ministry of Information and Communication (MIC) 21st Century Frontier R&D Program in Korea and by the department specialization fund, 2006 of the catholic university of Korea.

## References

1. Strang T., Linnhoff-Popien C., Frank K., CoOL: A Context Ontology Language to enable Contextual Interoperability, LNCS 2893: Proceedings of 4th IFIP WG 6.1 International Conference on Distributed Applications and Interoperable Systems (DAIS2003)
2. Gu, T., Wang, X. H., Pung, H. K., Zhang, D. Q. Ontology Based Context Modeling and Reasoning using OWL. In Proceedings of the 2004 Communication Networks and Distributed Systems Modeling and Simulation Conference (CNDS2004), San Diego, CA, USA, January 2004
3. Motik B., Oberle D., Staab S., Studer R., Volz R., KAON Server Architecture, Technical Report 421, University of Karlsruhe, Institute AIFB, 76128 Karlsruhe, Germany, 2002
4. SWAT Projects, HAWK - OWL Repository and Toolkit, <http://swat.cse.lehigh.edu/projects/index.html#hawk>
5. SungTae Kim, KyengWhan Jee, Jung-Jin Yang, Utilizing Distributed Ontology Repository in Multi-Agent System Environment, Journal of Intelligent Information Systems, Vol.11 (3) pp. 129-139, 2005
6. Jarrar M., Demey J., Meersman R., On Using Conceptual Data Modeling for Ontology Engineering. In Aberer K., March S., and Spaccapietra S., (eds): Journal on Data Semantics, Special issue on "Best papers from the ER/ODBASE/COOPIS 2002 Conferences", Vol. 2800, pp.:185-207, LNCS, Springer, ISBN: 3-540-20407-5, October 2003



# Object Recognition Using K-Nearest Neighbor in Object Space

Jong-Min Kim<sup>1</sup>, Jin-Kyoung Heo<sup>2</sup>, Hwan-Seok Yang<sup>1</sup>, Mang-Kyu Song<sup>1</sup>,  
Seung-Kyu Park<sup>2</sup>, and Woong-Ki Lee<sup>1</sup>

<sup>1</sup>Computer Science and Statistic Graduate School, Chosun University, Korea

<sup>2</sup>Division of Cyber Investigation Police, Howon University, Korea  
mrjjoung@chosun.ac.kr

**Abstract.** Object recognition technologies using PCA(principal component analysis) recognize objects by deciding representative features of objects in the model image, extracting feature vectors from objects in an image and measuring the distance between them and object representation. Given frequent recognition problems associated with the use of point-to-point distance approach, this study adopted the K-nearest neighbor technique(class-to-class) in which a group of object models of the same class is used as recognition unit for the images inputted on a continual input image. However, we propose the object recognition technique new PCA analysis method that discriminates an object in database even in the case that the variation of illumination in training images exists. Object recognition algorithm proposed here represents more enhanced recognition rate to change of illumination than existing methods.

## 1 Introduction

Object recognition is one of the most actively researched areas in computer vision [1]. An object recognition system can be described in various ways, but it simply finds objects in a given image that match models of known objects in the database [2][3]. Under this definition, the object recognition system responds differently in the presence of clutter: If there is almost no amounts of clutter in the image, a single object exists, and the system will find out whether the detected image matches an object representation in the database. In high clutter environments, the system will not only find out whether detected object matches the models in the database but also identifies characteristics features of objects such as area in the image.

In this study, a collection of images was developed by rotating a 3D object 5 degrees at a time and making a full turn. The performance of recognition systems using principal component analysis is very sensitive to rotation, translation, scale and illumination [4][5]. It is therefore necessary to create many images of objects to be tested and normalize the size of images to keep the performance of recognition system stable. Because of frequent recognition errors involved in the use of point-to-point approach[6], this study used k-Nearest neighbor approach (class-to-class), in which a group of object models of the same class is used as recognition unit for images inputted on a continual basis, to improve the recognition quality. Normalization and histogram equalization were also performed in object images to ensure a stable recognition rate under varying illumination conditions.

## 2 Object Recognition Algorithm

Normalization was applied to visual images obtained on a real time basis, and recognition was performed using principal component analysis. A local eigenspace was designed to apply the K-Nearest neighbor decision rule rather than simple distance measures. The proposed algorithm is presented in Fig 1.

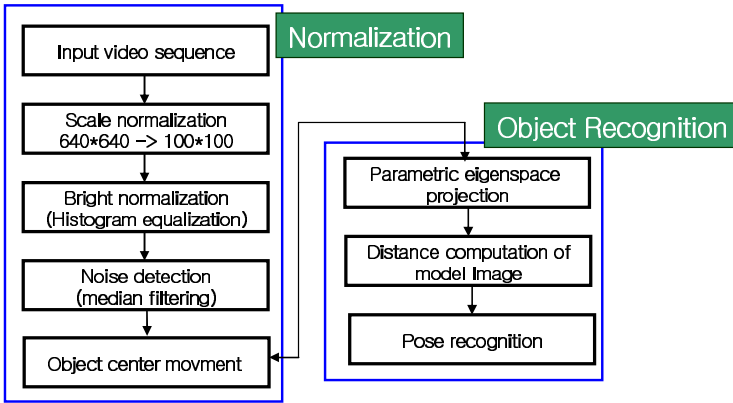


Fig. 1. Composition of proposed algorithm

## 3 Object Recognition Using PCA

This study proposes a solution based on principal component analysis to obtain images of object that is rotating a certain degree at a time and making a full turn

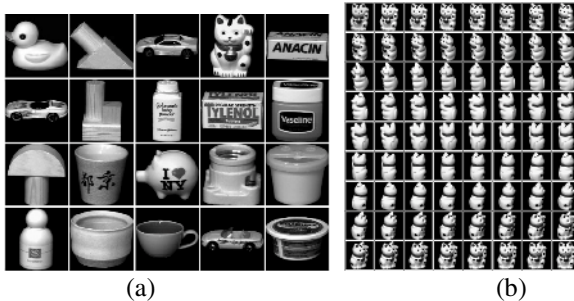


Fig. 2. (a) Object Set (b) Image set obtained by rotating object 5°

The solution involved creating a low dimensional vector space to model the overall appearance of object using the equalized output. The images of objects used for this study is presented in Fig. 2(a) and 72 images of an object that rotated 5° degrees at a time is presented in Fig. 2(b). In order to calculate eigenvectors, the average image

was calculated as the center value and then the difference from this center to each image was calculated using the following formulas (1) and (2):

$$C = (1 - N) \sum_{i=1}^N x_i \tag{1}$$

$$X \overset{\Delta}{=} \{ x_1^{(1)} - c, x_2^{(2)} - c, \dots, x_R^{(p)} - c \} \tag{2}$$

Where  $C$  is average image and  $X$  is a set of images.

The image matrix  $X$  is  $N \times M$ , where  $M$  is the total number of images in the universal set, and  $N$  is the number of pixels in each image.

Next, we define the covariance matrix :

$$Q = XX^T \tag{3}$$

That is, eigenvalue  $\lambda$  and eigenvector  $e$  of the covariance matrix  $Q$  of the images were calculated according to the following equations:

$$\lambda_i e_i = Q e_i \tag{4}$$

Among the studied matrixes, the matrix used as eigenvector is  $U$  as it size equals to matrix  $X$ . Eigenvectors yielded from the process of SVD(Singular Value Decomposition) can be reconstructed in the order of descending with eigenvalues. An eigenvector's eigenvalue reflects the importance of eigenvector and is calculated using the formula(5).

$$\frac{\sum_{i=1}^k \lambda_i}{\sum_{i=1}^N \lambda_i} \geq T_1 \tag{5}$$

Where  $T_1$  is the threshold value used for determining the number of vectors.

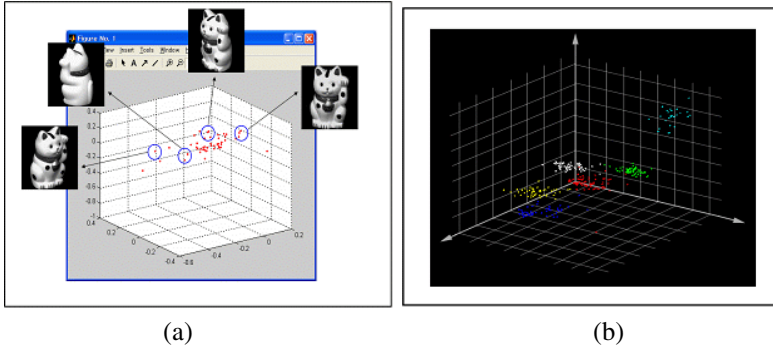
$K$  were 5 for the low dimensional space used for learning and pose evaluation.

### 3.1 Image Correlation and Distance in Eigenspace

Once object models are decided by normalized images in the eigenspace, the next step needed for recognition is very simple. After the difference between the input mage  $X$  and the average image  $C$  was calculated and reflected in the eigenspace according to the following formula:

$$f_j = [e_1, e_2, e_3, \dots, e_k]^T (x_n - c) \tag{6}$$

The new image  $f_j$  was represented as a point in the eigenspace. When reflected, a set of scattered points in the eigenspace correspond to all the objects.



**Fig. 3.** (a) The distribution of cat images in the eigenspace (b) The distribution of images of all objects in eigenspace

The closer distance between the fixed point to a point in the eigenspace is interpreted as a higher correlation between the new image and the previously stored image.

### 3.2 Distance Measures and Object Recognition with Improved k-Nearest Neighbor

The use of point-to-point approach in pattern classification frequently resulted in a recognition error even if a new image was matched to the previously stored image because of features that were unnecessarily detected in the new image. To solve this recognition problem associated with this individual feature matching technique, features can be analyzed by using a group of object models of the same class as recognition unit for images inputted on a continual basis (Class to Class).

$$w = \frac{(\arg S(M_j) - \text{Min}(\arg S(M_j)))}{d(k - 1)} \tag{7}$$

Where  $\arg S(M_j) = j$  is an operator meaning the number of object model.  $K$ -Nearest Neighbor matching algorithm was used as presented in formulas (7) and (8).

$$\frac{\sum \sum w(I_j - M_j)}{k} \tag{8}$$

Where  $K = 3$ . Recognition of model image and the input image is decided by the value obtained from the formula (8). Based on these formulas, input images were matched to model images as illustrated in the eigenspace(Fig. 4).

It was found in the same eigenspace that the input image can be identified as a different object despite its proximity to the model image. To solve this recognition problem resulted from the distance measures between points, this study used  $K$ -Nearest Neighbor matching algorithm in which features are analyzed for matching by using a group of object models of the same class as recognition unit for images inputted on a continual basis. As a result, Recognition quality improved.

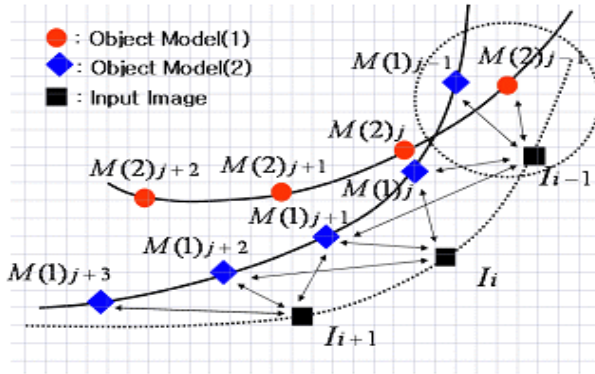


Fig. 4. Recognition based on  $K$ -Nearest Neighbor matching algorithm

## 4 Experiment Results and Conclusions

The images were taken while each object was rotating  $5^\circ$  at a time and making a full turn. A set of these images is called the image of the object. The images in the size of  $640 \times 480$  pixels were normalized to the size of  $100 \times 100$  pixels. Eigenvectors were calculated from a set of object images, and the five-dimensional vectors showing high probabilities were defined as a feature space. Matching rates of point-to-point approach and edited  $K$ -nearest neighbor rule are compared in Table 1. As shown in the table, the matching rates were high with edited  $k$ -nearest neighbor rule. A larger number of mismatches were corrected when using the  $k$ -nearest neighbor rule.

Table 1. Comparison of matching rates between two classification techniques

Matching	input image	A failure to find a match	Mismatching	Matching rate
Distance measure (Point to Point)	With object models	10.5 %	11 %	78.5 %
	Without object models	15.8 %	20.2 %	62 %
K-Nearest Neighbor (Class to Class)	With object models	6.1 %	3.7 %	90.2 %
	Without object models	13.2 %	16.8 %	70 %

Effects of varying illumination conditions on recognition rates are presents in Fig.5. Illumination conditions were defined as the unaltered illumination condition (simple principal component analysis), normalized brightness and the condition after histogram equalization.

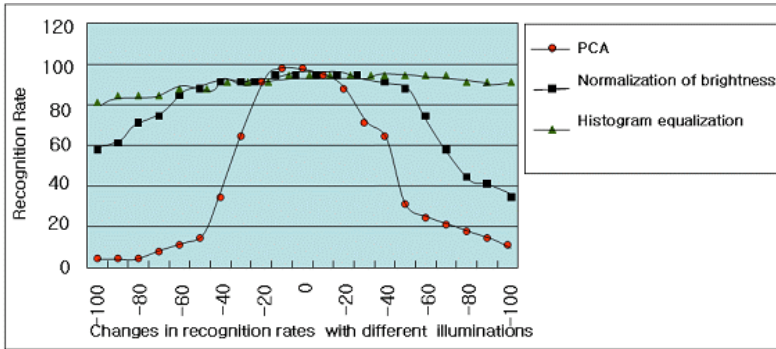


Fig. 5. Comparison of recognition

It was found that matching rates were high when features were classified with k-nearest neighbor rule, compared with those obtained from the use of point-to-point measures. The study findings provided the evidence that k-nearest neighbor was more effective for simple and stable recognition than other techniques using geometric information or stereo images. It is known that recognition systems using principal component analysis undermines recognition quality because of their vulnerability to changes in illumination conditions. The edited K-nearest neighbor rule proposed in this study maintained recognition rate of more than 90% under varying illumination conditions caused by histogram equalization, and the recognition rate was higher than that obtained from the illumination condition in which brightness was normalized.

## References

1. J.Weng, N.Ahuja, and T.S.Huang, "Learning recognition and segmentation of 3-D object from 2-D images." Proc. of Fourth Int'l Conf. on Computer Vision, pp. 121-128, Belin, May 1993.
2. Paul Viola, M. Jones, "Robust real-time object detection", *International Conference on Computer Vision*, 2001.
3. Hiroshi Murase and Shree K. Nayar, "Visual Learning and Recognition 3-Object from appearance", *international journal of Computer Vision*, Vol,14,1995.
4. Daisaku Arita, Satoshi Yonemoto and Rin-ichiro Taniguchi. Real-time Computer Vision on PC-cluster and Its Application to Real-time Motion Capture. 2000 IEEE.
5. J. Yang, D. Zhang, "Two-Dimensional PCA: A New Approach to Appearance-Based Face Representation and Recognition," *IEEE Transactions on Pattern analysis and Machine Intelligence* Vol. 26, No. 1, 2004. 1.
6. F. Bourel, C.C Chibelushi and A.A Low, "Robust facial expression recognition using a state-based model of spatially localised facial dynamics", *Proceedings of Fifth IEEE International Conference on Automatic Face and Gesture Recognition*, pp.106-111, 2002.
7. Hwan-Seok Yang, Jong-Min Kim, and Seoung-Kyu Park, "Three Dimensional Gesture Recognition Using Modified Matching Algorithm", *Lecture Notes in Computer Science LNCS3611 pp224-233*, 2005

# Research on Smart Multi-agent Middleware for RFID-Based Ubiquitous Computing Environment\*

Minwoo Son, Joonhyung Kim, Dongil Shin<sup>\*\*</sup>, and Dongkyoo Shin

Department of Computer Science and Engineering, Sejong University  
98 Kunja-Dong, Kwangjin-Ku, Seoul 143-747, Korea  
{minwoo15, joon1979, dshin, shindk}@gce.sejong.ac.kr

**Abstract.** Previous RFID (Radio Frequency Identification) middleware did not include intelligent capabilities such as automatically controlling home appliances and users. Therefore, we add USN (Ubiquitous Sensor Network) technology, which recognizes and manages an appliance's state-information (temperature, humidity, pollution and so on) by connecting RFID tags, and propose the Smart Multi-Agent Middleware architecture for intelligently managing the RFID-based ubiquitous computing environment. We also present the proposed architecture's execution procedure for controlling ubiquitous appliances.

**Keywords:** Ubiquitous Computing, RFID(Radio Frequency Identification) USN(Ubiquitous Sensor Network), Multi-Agent, Middleware.

## 1 Introduction

'Ubiquitous Computing' [1] means that users can utilize computers naturally and conveniently as a part of their daily routine, regardless of place and time. It means that a computer existing anywhere can use specialized services, changing its contents according to place or time via sensing and tracking. This means that the efficiency of users' lives can be maximized through a ubiquitous network.

However, it is difficult to apply various ubiquitous appliances in current ubiquitous computing because IP addresses must be assigned to every appliance in the ubiquitous computing environment. It is too labor intensive to find proper appliances to communicate if a wireless network is used in ubiquitous computing. Furthermore, a ubiquitous computing environment must be modified with the addition of new appliances; it is difficult to implement modules for TCP/IP communication.

To solve these issues, RFID [2,3,4,5] has been applied to the ubiquitous computing environment in recent years. RFID technology describes the use of radio frequency signals to provide automatic identification of items. RFID is a flexible technology that is convenient, easy to use, and well-suited for automatic operation. RFID can be supplied as read-only or read/write, does not require contact or line-of-sight to operate, can

---

\* This study was supported by a grant of the Korea Health 21 R&D Project, Ministry of Health & Welfare, Republic of Korea. (0412-MI01-0416-0002).

\*\* Correspondence author.

function under a variety of environmental conditions, and provides a high level of data integrity. Because of these advantages, RFID is widely used in various areas of daily life. For example, RFID middleware technologies such as Singularity[4], which is a suite of RFID middleware to support RFID enabled Supply Chain Management, enterprise integration, and EPCglobal [5] for managing data streams, have progressed in several projects. Because RFID middleware technology such as Singularity and EPCglobal is only able to control ubiquitous appliances in an RFID-Based environment, Singularity and EPCglobal projects emphasize the necessity of research on smart technologies that allow users to efficiently and easily control appliances. Future work will involve smart middleware for RFID-based ubiquitous computing environments.

We add USN [6] technology, which recognizes and manages an appliance's state-information (temperature, humidity and so on) by connecting RFID tags, to solve RFID's weakness, and propose a Smart Multi-Agent Middleware for efficiently managing the RFID-based ubiquitous computing environment. The efficiency of daily life using a ubiquitous computing environment will be maximized through a ubiquitous computing environment that has Smart Multi-Agent Middleware that is able to learn on its own.

## 2 Design of RFID-Based Ubiquitous Computing Environment Architecture

It is difficult to apply various ubiquitous appliances in current ubiquitous computing because IP addresses must be assigned to every appliance in the ubiquitous computing environment. Therefore, to solve these problems and to efficiently manage ubiquitous appliances for users, we designed an RFID based ubiquitous computing environment architecture, as shown in figure 1.

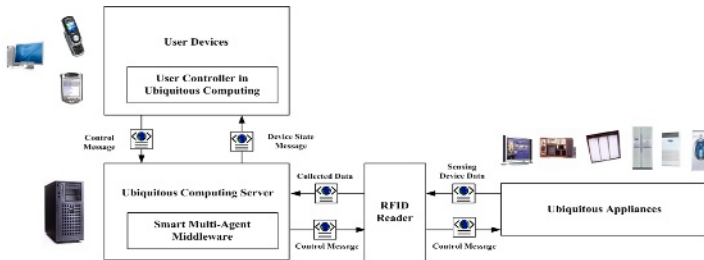


Fig. 1. Architecture of RFID-based Ubiquitous Computing Environment

This figure is the design for an architecture to efficiently control RFID-based appliances in a ubiquitous computing environment. To control ubiquitous appliances, a user utilizes two connection-systems. The first method makes it possible for a user to control ubiquitous appliances, after the user is authenticated through a web browser that uses a web service. The second method is an HIML (Human Interaction Markup Language) [7] document that is transmitted using a Mobile Device, such as a PDA or Web PAD, using a Network Service. RFID sensors, which are attached to all



appliances in order to sense the state of the appliances, periodically send the state of the ubiquitous appliances. This ubiquitous appliance state-information is transferred to RFID readers. The Smart Multi-Agent Middleware collects this data from the RFID readers, analyzes the data, and determines the operations of ubiquitous appliances in order to provide an optimal environment to the user of the ubiquitous computing environment. The Smart Multi-Agent Middleware sends control messages to the RFID readers after it determines the operations. RFID readers transfer the control messages received from the ubiquitous server to the ubiquitous appliances. The ubiquitous appliances perform operations according to the control messages.

### **3 Smart Multi-agent Middleware for RFID-Based Ubiquitous Computing Environment**

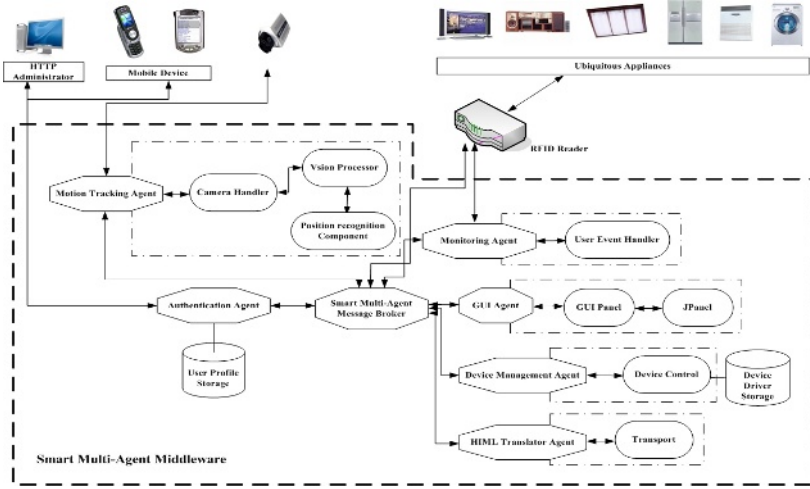
Lately, active research has been taking place on projects involving RFID middleware, which supports the control of appliances for users in RFID-based environments. However, although RFID Middleware technology, such as Singularity, EPCglobal, and so on, is able to control home appliances in an RFID-Based environment, the agents don't support smart technology. This means that the user can't efficiently and easily control appliances in the ubiquitous computing environment. To solve this problem, Singularity and EPCglobal projects emphasize the necessity of research on smart technologies that support the ability of users to efficiently and easily control appliances. Future work will involve smart middleware for RFID-based Ubiquitous Computing environments.

Therefore we propose Smart Multi-Agent Middleware, which supports the efficient control of ubiquitous appliances by users, for an RFID-Based Ubiquitous Computing Environment. Accordingly, the proposed agent, which can autonomously control the ubiquitous computing environment instead of the user, is necessary in order to save the user time and effort through the Monitoring Agent.

#### **3.1 Architecture of Smart Multi-agent Middleware**

Since all the control functions for a home appliance are defined in the Smart Multi-Agent Middleware, if an agent wants to control a particular home appliance in a home network, it has to look-up the control variables for the target ubiquitous appliance in the Smart Multi-Agent Middleware. The Smart Multi-Agent Middleware can then invoke the control functions using a remote control method invocation. The following is the execution procedure to control a ubiquitous appliance.

As shown in figure 2, to control ubiquitous appliances, a user uses two connection systems in the Smart Multi-Agent Middleware. The first method makes it possible for a user to control a ubiquitous appliance, which has an attached sensor, in the Smart Multi-Agent Middleware, after the user is authenticated through a User Authentication Agent, which utilizes SSO (Single Sign-On) [8], in a web browser. The second method uses an HIML document transmitted using a mobile device such as a PDA or Web PAD through a User Authentication Agent, using a network (sensing, TCP/IP and so on) for an HIML Translator approach, which after parsing approaches the Smart Multi-Agent Message Broker.



**Fig. 2.** Architecture of Smart Multi-Agent Middleware

To control a ubiquitous appliance through a web or mobile device, it accepts data on access privileges by a user device in a User Profile Storage through a user ID if users approach. After the Smart Multi-Agent Message Broker receives a ubiquitous appliance ID and appliance function-services, if finds an appropriate driver through the Device Driver Storage and appliance information through the Device Management Agent.

The Smart Multi-Agent Middleware observes the states of the ubiquitous appliances through the Monitoring Agent. An agent that can autonomously control the ubiquitous computing environment instead of the user is necessary in order to save the user time and effort. This agent must continuously collect data from the RFID Tags, which comes from the RFID sensors, analyze the collected data, and find out the inclinations of the user, based on the analyzed information. If the User Event Handler finds an appliance event, it solves the appliance event. If the User Event Handler doesn't solve the event, after the User Event Handler sends the appliance's status information to the Smart Multi-Agent Message Broker, the Smart Multi-Agent Message Broker solves the appliance event.

When recent RFID middleware receives data from the Tags of multiple appliances (for example, when a user is in a position between an audio source and a TV), the middleware generates an event or supports the services of the multiple appliances.

However, the proposed Smart Multi-Agent Middleware solves problems like this through a Motion Tracking Agent. When the Smart Multi-Agent Middleware recognizes a user's location through a Motion Tracking Agent, after the Smart Multi-Agent Middleware receives data from the appliances through an RFID Reader, the Smart Multi-Agent Middleware supports the service of the closest appliance to the user's location through the Motion Tracking Agent.

We here briefly introduce the services of the Agent portion of the Smart Multi-Agent Middleware. The Smart Multi-Agent Message Broker implements a generic

agent functionality for sending and receiving messages. At the system level, this could be anything that would respond to commands (a light switch, a washing machine, and so on). An HIML Translator Agent facilitates the automatic translation of HIML strings into internal message objects. The Device Manager Agent manages each ubiquitous appliance’s driver and service (addition, insert and remove) and periodically updates the driver and stores the driver in driver storage. The Monitoring Agent observes each appliance and logs the usage information for each appliance. In addition, if an appliance causes an event, the Monitoring Agent solves the event through the User Event Handler. The Motion Tracking Agent sends the recognized user’s location to the Smart Multi-Agent Message Broker.

### 3.2 Sequence Descriptions in Smart Multi-agent Middleware

Figure 3 shows the process of posting a message from one Agent to another Agent. Sending requires a Destination and a MessageBody object. The Transport then constructs a Message object, and sends it (via Multicast/UDP socket) to its intended recipient(s). The receiving Agent will typically issue a Receive() call to its Transport object. The Transport object sets up a receive buffer and a receive packet, and then waits for an incoming packet. Once a packet arrives, it is converted into an HIML string. The HIML string is then converted into a Message object via the HIML Translator Agent owned by the Transport. The Transport then uses that returned Message object as a return value for its Receive() method. Finally, the receiving Agent passes the Message object to its processMessage() method, which is an abstract method that needs to be implemented by all Agent-derived classes.

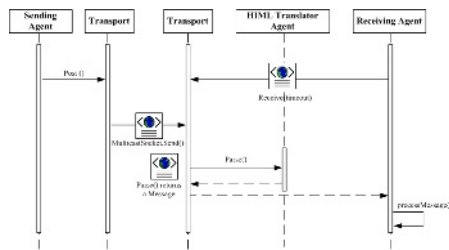


Fig. 3. Sequence diagram of Messaging

Figure 4 shows the process followed during GUIAgent startup. The first thing that the GUIAgent does is to call the setUpLocalPanels() method. The application must implement an application-specific version of GUIAgent (derived from GUIAgent), which provides the setUpLocalPanels() method. In the case of the RFID application, setUpLocalPanels() will set up the Admin and Test panels in the GUI. After setting up the local panels, the GUIAgent will query the Backbone for a list of currently connected Agents. The GUIAgent will then attempt to create and add GUIPanels that correspond to each of the Agents in the list. Finally, the GUIAgent will list the ConnectedAgent messages, adding new GUIPanels.

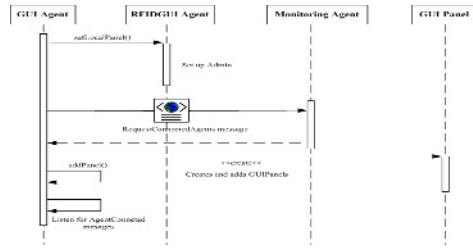


Fig. 4. Sequence diagram of GUI Agent Startup

## 4 Conclusion and Future Direction

This paper proposes Smart Multi-Agent Middleware for an RFID-Based Ubiquitous Computing Environment.

Smart Multi-Agent supports the efficient control of ubiquitous appliances for users, for an RFID-Based Ubiquitous Computing Environment. Accordingly, an agent that can autonomously control the ubiquitous computing environment instead of the user is necessary in order to save the user time and effort through the Monitoring Agent. One of the most important future topics in our project involves a context awareness study for a Multi-Agent’s powerful automatic service system. Therefore, we will be doing a study using the Smart Multi-Agent Middleware to manage ubiquitous appliances, after extending its services to include things such as context awareness and authenticated and distributed security.

## References

1. Shulzrinne. H., Schulzrinne. H., Xiaotao. Wu, Sidiroglou. S., Berger. S.: Ubiquitous computing in home networks, Communications Magazine, IEEE, Vol. 41, Issue. 11, (2003) 128-135
2. Radio Frequency Identification (RFID) homepage, <http://www.aimglobal.org/technologies/rfid>
3. Finkenzeller Klaus : RFID Handbook : Radio-Frequency Identification Fundamentals and Applications in Contactless Smart Cards and Identification, (2004)
4. Singularity homepage, <http://singularity.firstopen.org/>
5. EPCglobal homepage, <http://www.epcglobalinc.org/index.html>
6. Ministry of Information and Communications Republic of Korea, The Basic Plan of U-Sensor Network Construction, (2004)
7. Gunhee Kim, Dongkyoo Shin, Dongil Shin : Design of a Middleware and HIML(Human Interaction Markup Language) for Context Aware Services in a Ubiquitous Computing Environment, EUC 2004, (2004) 682-691
8. Jongil Jeong, Dongukyoo Shin, Dongil Shin, Hyun-Mok Oh: A Study on the XML-Based Single Sign-On System Supporting Mobile and Ubiquitous Service Environments, Embedded and Ubiquitous Computing, International Conference EUC 2004, (2004) 903-913

# Certificate Management System in MANET for Ubiquitous Computing

Dae-Young Lee and Sang-Hyun Bae

Dept. of Computer Science & Statistics, College of Natural Sciences of Chosun University  
375 Seosuk-Dong, Dong-Gu, Gwang-ju, 501-759, Korea  
Tel.: +82-062-230-7962; Fax: +82-062-234-4326  
cssna01@chosun.ac.kr

**Abstract.** This study addressed security requirements for ad-hoc network environments, which lies at the heart of the ubiquitous computing revolution and proposed a partially-distributed certificate management system that can ensure security in mobile ad-hoc networks. The proposed model is characterized by its ability to handle dynamic mobility of nodes, minimize routing load and enhance expandability of network by allowing participating nodes to authenticate each other without being interrupted by joining the cluster. The security, efficiency and robustness of the proposed model were evaluated through simulation.

## 1 Introduction

Through ubiquitous computing for which computers are invisibly planted in daily life and things, information can be exchanged and we can use ubiquitous computers anytime and anywhere[1]. That is, ubiquitous computing is our daily environment in which we can not recognize the existence of computers they are incorporated into our daily life though computers are installed in our physical space consisting of things and environments. For existing security, information digitalized in cyber space or physical space and stored in computers was a problem. In consideration of properties of ubiquitous networks, a problem that personal information may be exposed exists in ubiquitous network. For existing networks, the place attacked was confined to personal computers, but in ubiquitous networks, all the personal space may be revealed. Therefore, cyber terror in ubiquitous network includes terror given to physical space, things and physical bodies and extensive space protection is required beyond protection of personal, business and national information.

One of the advantages of Mobile Ad-Hoc Network is the independence from the centralized administration, enabling users to access to the network freely and flexibly. Security support for MANET poses a challenge for system developers because of dynamic network reconfiguration caused by independent nodes [2][3][4].

Since authentication is performed combining  $k$  number of partially distributed certificates into a full certificate in public-key management system [2], the time required for the authentication process increases if the number of bits rose. In other words, increased packet data length causes network overhead and eventually undermines stability and efficiency of the network, increasing the risk for data eavesdropping or alteration.

This study suggests security requirements for MANET which can be a base of ubiquitous system and models that can prevent security threat through application in MANET. It is also solve excessive loading found in centralized control model by dispersing CA for adjustment to dynamic changes of nodes of MANET quickly and suggests a system model which supports expansion so that existing nodes performing communication within clusters can provide active certificate service without being affected by input of new nodes. In addition, it is to evaluate its stability, effectiveness and strength through simulation of the suggested model.

## 2 Related Work

### 2.1 Secret Sharing

Under secret sharing scheme, participating nodes hold a share of secret key as a means of authorization to participate in authentication process. There are one dealer and  $n$  number of players. The dealer allocate a share of secret key to each player when the player meets specific requirements, and the authentication can be performed when at least  $t$  share holders combine their shares. This is, a set of less than  $t$  share holders is not able to complete authentication.

Cluster head uses public/shared secret key pair and encrypts data using the public key and unicasts encrypted data to its neighboring nodes. Each receiving node forwards its share of secret key to trusted nodes through a safe channel. Once more than  $t+1$  valid share holders combine their shares, the completed secret key is transmitted to each receiving node.

### 2.2 Threshold Cryptography

In a public-key management system, public key cryptography design is the most important to make user's public key trustful. The absence of trusted authentication authority in MANET is cited as an obstacle for using a self-organized public-key management system for MANET. As an alternative to a centralized authentication authority, the use of threshold cryptography backed by share refreshing techniques becomes commonplace [5].

### 2.3 Share Refreshing

A mobile attacker or invader is deemed as a virus within a network as it temporarily attacks a server and keeps moving in search for another victim, meaning all servers are can be attacked in the end. It is possible for the attacker to collect partially distributed secret key beyond the threshold size  $t$  even when severers detect the attack and prevents further invasion [6][7]. It may be however feasible for an attacker to generate a valid certificate if it already obtains enough shares of secret key. To address this security problem, share refreshing can be employed [8].

The share refreshing algorithm used for the threshold cryptography scheme calculates a new share of secret key from the old share without the need for receiving a new share from sever. A new sharing  $(n, t+1)$  is then performed to share refreshed secret key. After share refreshing, each sever eliminates its old share and uses

refreshed share to generate a certificate. It is therefore impossible for the attacker to use old shares it has to authenticate a certificate signed with refreshed share, making the network safe from attack.

## 2.4 Self-initialization

Localized self-initialization algorithm is employed to ease the job of secret share dealing among nodes. Under this algorithm, a participating node holding a share of the secret key is able to initialize a sharing process for new nodes in the absence of a trusted dealer. And new nodes form a cluster based on shared secret key.

## 2.5 Cross Certification

Cross certification is an authentication method undertaken by certification authorities managing different PKI domains to establish a trust relationship. A cross certification process is undertaken between two certification authorities representing its own PKI domain. The top-level certification authority in a hierarchy of certification authorities is Root CA. The process of cross-certifying CA is not always possible with the top level CA, but it can be undertaken other CAs in the hierarchy.

# 3 System Design

Once the cluster is formed, the cluster head acts as a certification authority for all its members and is responsible for establishing a new pair of public/secret keys to be used for signing certificates. The cluster head will unicast its self-generated public key to other cluster heads through a network backbone. Thus the cluster heads hold the shares of public key.

The cluster head serving as certification authority verifies the feasibility of secret sharing using the cross-certification relationship in place when it is aware of neighboring cluster members ( $k$ ). After the sharing process is verified, the cluster head unicasts secret share to other cluster heads for their own verification. If the verification fails, the sensor node requests the requesting cluster head to retransmit the secret share. Once the secret share is successfully verified using random numbers, the sensor node generates a share of secret key. Each cluster member generates a partial certificate using shared secret key. The cluster head will request its neighboring cluster members to present their own certificates to sign public key certificates. Each partial certificate is transmitted to the requesting node through cluster head. When the number of partial certificates exceeds the threshold size, the requesting node can combine partial certificates together and create a complete certificate.

## 3.1 Creation of Public Key and Secret Key

The cluster head that assumed the role of CA (called CA node hereinafter) generates random numbers and a partial public key value ( $PK_i$ ) and unicasts its self-generated public key to other CA nodes through a network backbone. After partial public key is delivered to individual CA nodes, each CA node issues its own domain's public key in the following formula and keeps the key.

$$PK = \prod_{i=1}^n PK_i \tag{1}$$

If there are 4 clusters in the network, each CA will receive 4 different values (PK<sub>1</sub>, PK<sub>2</sub>, PK<sub>3</sub> and PK<sub>4</sub>) as public key. CA nodes will generate K-1 polynomial (f<sub>i</sub>) with a random number of x<sub>i</sub> and y intercept when they are aware of k number of neighboring nodes. To verify the feasibility of secret sharing, a CA node sends control information (f<sub>ij</sub>) about polynomial (f<sub>i</sub>) having coefficient of f<sub>ij</sub> to other CA nodes to perform cross certification.

$$f_i = f_{i0} + f_{i1}x^1 + f_{i2}x^2 + \dots + f_{iK-1}x^{K-1}, f_{i0} = x_i \tag{2}$$

$$F_{ji} = g^{f_{ij}}, (j = 0, \dots, k - 1)$$

CA nodes unicast the value of secret share (SK<sub>ij</sub>=g<sup>f<sub>ij</sub></sup>) corresponding to self-generated random numbers to other CA nodes. When secret is delivered to CA nodes, each CA node verifies SK<sub>ij</sub> value. If the value is not verified, the sensor node requests the retransmission of SK<sub>ij</sub> value. Once shared secret is verified, each CA node will generate a share of secret key using the following formula.

$$SK = \prod_{j=1}^n SK_{ji} \tag{3}$$

### 3.2 Distribution of Secret Share

Cluster heads serving as CA in MANET are responsible for coordinating key distribution among cluster members. Cluster’s public key is delivered to cluster members through the procedures described in previous paragraphs. Once each cluster head establishes a pair of public/secret keys, the cluster head distributes shares of secret key to k nodes within the cluster. A certificate is issued based on a close cooperation between cluster members, and partially distributed secret key is used to sign public key certificate. In the proposed model, CAs are widely distributed to support for the dynamic nature of the network in which nodes join or leave the trusted administrative domain rapidly and randomly. As a result, the partially distributed certificate authority scheme is characterized by its mobility and flexibility.

### 3.3 Certificate Creation

Although secret key is partially distributed among n nodes within a cluster, a partial certificates held by k nodes will be required to issue a certificate that matches the one issued by a CA.

The proposed model allows CA node to distribute self-generated secret key to k nodes, and a node can create a certificate in collaboration with neighboring nodes within the threshold transmission range without rely on the central certification authority. Secret key shared among all cluster members will be demanded to sign public key certificate with new nodes joining the cluster.



Each cluster member is capable of generating a partial certificate based on secret sharing scheme. A cluster member will broadcast its request for partially distributed certificate to all its neighbors to sign public key certificate and create a new certificate. Partial certificates will be forwarded to the requesting node through cluster head. Partial certificates are then combined into a complete certificate when the threshold size is met. A certificate is created according to the following process.

Secret sharing among all nodes within a cluster is based on a secret polynomial  $f(x)$ . Security of the ad-hoc network in which public/private keys  $(SK=\langle d,n \rangle / PK=\langle e,n \rangle)$  are shared among  $k$  nodes can be achieved by constructing a Lagrange interpolation polynomial (1) of degree  $K-1$  as follows:

$$f(x) = d + f_1 \cdot x + f_2 x^2 + \dots + f_{K-1} \cdot x^{K-1}, \quad N(> K) \tag{4}$$

However, when the number of nodes is less than  $k$ , secret key information cannot be shared. Shared secret key is expressed as  $f(0)=d$ , and the value of secret share held by node  $i$  is expressed as  $P_i=f(C_i) \bmod n$ .

Each node can sign certificates using shared secret key. Shared secret key  $(SK=\langle d,n \rangle)$  is obtained by calculating  $d$  after collecting  $k$  secret shares in the following formula:

$$\begin{aligned} d &\equiv \sum_{i=1}^K (P_{C_i} \cdot l_{C_i}(0) \bmod n) \\ &\equiv \sum_{i=1}^K SK_i \pmod n \end{aligned} \tag{5}$$

Where  $l_{C_i}(0)$  represents Lagrange coefficient.  $SK=\langle d,n \rangle$  can be performed by a coalition of more than  $k$  nodes using signed partial certificates.  $sk_{ij}$  can be recovered from polynomial  $(d)$  in equation (6), which was derived from the equation (4).

$$\begin{aligned} d &\equiv \sum_{i=1}^K (P_{C_i} \cdot l_{C_i} \bmod n) \\ &\equiv t \times n + d \end{aligned} \tag{6}$$

Where  $(P_{C_i} \cdot l_{C_i} \bmod n)$  represents a module concerning  $n$  and its value equals to  $d$  of the formula (6). It would be difficult to create a valid certificate by multiplying  $k$  partial certificates. A complete certificate can be created using the following  $K$ -bounded coalition offsetting algorithm.

$$M^{t \times n + d} \equiv M^{t \times n} \times M^d \equiv M^d \pmod n$$

Where  $M$  is public key  $\langle e,n \rangle$ . The threshold  $K$  represents the number of nodes within a cluster. And the threshold size is not required to be large.

A certificate is issued through collaboration among nodes and shared secret key serve as certificate signing key. Under the secret sharing scheme used in this study, certification authorities are widely distributed in keeping with dynamic mobility of nodes that rapidly join and leave trusted administrative domains. As a result, the probability of a successful key agreement is high due to its mobility and flexibility.

## 4 Simulation for Testing and Evaluation

Network Simulator 2 (NS2) written in C++ with OTcl interpreter, an object-oriented version of TCL, was used to evaluate the performance of the proposed certificate management mechanism. Packet flows were traced using timing and the amount of routing information. And results were compared with those obtained from the use of existing routing protocols. Efficiency, security and robustness of the proposed certificate management system are evaluated as follows:

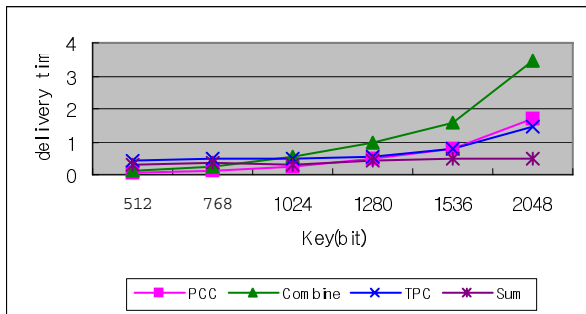
### 4.1 Efficiency and Security of the System

PCC represents computation time required to issue a partial certificate using secrete key. Combined time represents calculation time required to issue a complete certificate by combining k partial certificates[2]. TPC indicates computation time for a partial certificate based on Lagrange interpolation. Sum indicates total time required to combine k partial certificates, obtain secrete key and issue a certificate.

The results revealed that PCC and Combine computation time rose as the number of bits increased, implying an increase in routing overhead but a decrease in security effectiveness of the ad-hoc network. The efficiency and security of the network can be undermined as a result, and the risk of losing data or exposing confidential data grows.

However, computation time in the proposed model (TPC and Sum) slightly increased with increasing the number of bits, the pace of increase was much slower than that of PCC and Combine. Total computation time (SUM), particularly, maintained a rather steady delivery time despite increase in the number of bits. The efficiency and security of the network is sustained despite the increase in packet data length.

Changes in computation time for a partial certificate and a complete certificate were evaluated in relation to various k values. It is found that the variable k did not have an impact on the efficiency of the proposed model because existing member nodes steadily perform individual calculation for a partial certificate. Since the node requesting partially distributed certificates performs the whole process involving certificate creation, unnecessary system overhead does not incur.

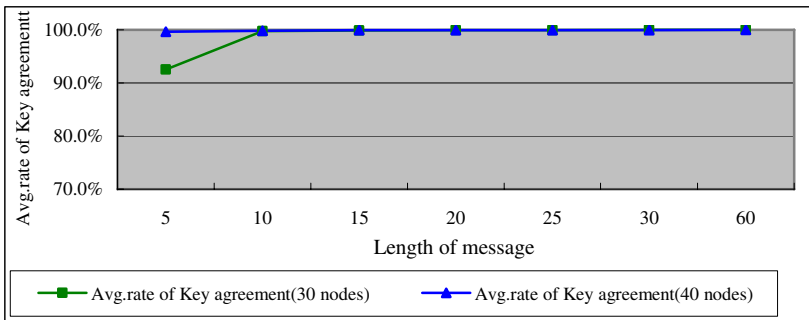


**Fig. 1.** Comparison of changes in computation time with the number of bits between TPC and PCC, key: 1024bit, Pentium III/500 laptop

## 4.2 Robustness

The robustness of the network was also evaluated by measuring a mutual key agreement between packet data and shared key-information message and changes in routing overhead with the number of nodes. The agreement between routing message and secret key was measured by comparing changes in number of nodes and the length of message (packet) sent and received by each node.

The key agreement among 30 and 40 k nodes was illustrated in Fig 2. The average rate of key agreement indicates the packet delivery ratio and the length of message indicates the packet size.



**Fig. 2.** Avg. rate of Key agreement (node  $K=30, 40$ )

The key agreement was maintained at an average rate of approximately 100% even after the number of nodes increased to 40 from 30. At the same time, key agreement was not affected by the increase in the length of message, meaning that packet data transmitted for the establishment of secret key by CA and shared key-information message are perfectly matched. The algorithm that allows nodes to forward and receive packet data on a stable basis does not incur unnecessary routing overhead, helping maintain robustness of the network.

## 5 Conclusion

Certificate creation time rose as the number of bits increases under existing certificate-based authentication protocol in which  $k$  partial certificates are generated using shared secret key and combined into a complete certificate. The increase in packet size increases the routing of the network traffic but decreases efficiency and security of the network, making routing data more susceptible to misinformation attacks. In the proposed model, certificate creation time slightly rose as the number of bits increased. But the pace of increase was much slower than that obtained from the use of existing certificate-based authentication protocol. In addition, the proposed model offered a steady delivery time in the certificate creation phase despite the increase in packet size. The efficiency and security can be therefore maintained in the network.

It was also found that the efficiency of the network was not influenced by changes in number of nodes ( $k$ ) because partial certificates are consistently generated by a

coalition of existing member nodes without being interfered by nodes joining the cluster. Since the node requesting partially distributed certificates performs the whole process involving certificate creation, unnecessary system overhead can be eliminated. This study suggests an ideal certificate management system for MANET by eliminating the need for pre-distribution of certificate signing key and centralized CA. Besides, the proposed system enhances the expandability of the network by incorporating joining nodes into certificate authority and allowing existing member nodes to perform the certification process steadily amid dynamic mobility of nodes.

## References

- [1] Mark Weiser. "The Computer for the 21st Century," *Scientific American*. 265:pp.94-104. 1991
- [2] J. Kong, P.Zerfos, H.Luo, S. Lu, L.Zhang. "Providing Robust and Ubiquitous Security Support for Mobile Ad-Hoc Networks", *IEEE Computer Society, Proceedings of the Ninth International Conference on Network Protocols (ICNP'01)*, PP. 251. 2001
- [3] Y. Zhang and W. Lee. "intrusion detection in wireless ad-hoc network." *ACM Mobicom*. Aug. 2000
- [4] L.Zhou and Z. Hass. "Securing ad hoc networks." *IEEE Network*. pp. 24-30. Nov/Dec. 1999
- [5] Y. Desmedt and Y. Frankel, "Threshold Cryptosystems", *Advances in Cryptology-Crypto '89*, (Lecture Notes in Computer Science 435), G. Brassard, Ed., Springer Verlag, May 1990, pp. 307-315
- [6] S.Jarecki, "proactive Secret Sharing and Public Key Cryptosystems," Master's Thesis, Dept. Elec. Eng. and Comp. Sci., MIT, Sept. 1995
- [7] A. Herzberg et al., "Proactive Secret Sharing or: How to Cope with Perpetual Leakage," *Advances in Cryptology-Crypto '95* (Lecture Notes in Computer Science 963), D. Coppersmith, Ed., Aug. 1995, Springer-Verlag, pp. 457-469.
- [8] Y. Frankel et al., "Proactive RSA," *Advances in Cryptology - Crypto '97* (lecture Note in Computer Science 1294), Springer-Verlag, U.S.A., Aug. 1997, pp. 440-454

# FPGA Based Intrusion Detection System Against Unknown and Known Attacks

Dong-Ho Kang, Byoung-Koo Kim, Jin-Tae Oh, Taek-Yong Nam, and Jong-Soo Jang

Information Security Research Division  
Electronics and Telecommunications Research Institute  
161 Gajeong-dong, Yuseong-gu, 305-350 Korea  
{dhkang, bkkim05, showme, tynam, jsjang@etri.re.kr}

**Abstract.** Network intrusion detection systems often rely on matching patterns that are gleaned from known attacks. While this method is reliable and rarely produces false alarms, it has the obvious disadvantage that it cannot detect novel attacks. Accordingly, an alternative approach which can be a combination with pattern matching approach is needed. We have made effort to design and implement high speed protocol anomaly and signature based intrusion detection approach to detect known and unknown attacks. This approach extracts a set of service fields from the application payload where many attacks occur and analyzes the value of fields to verify attack. This approach is implemented on the FPGA (Xilinx Virtex II pro) device to process packet at gigabit-per-second data rates.

**Keywords:** Network Security, Intrusion Detection, Protocol Anomaly Detection.

## 1 Introduction

Signature-based intrusion detection systems simply monitor network traffic using a detection methods. IDS devices rely almost entirely on string matching and breaking the string match of a poorly written signature is trivial. Not all IDS devices are signature-based; however, most have a strong dependency on string matching. Consequently, Signature-IDS face the packet leaking problem with the increase in the network speed.

SGS (Security Gateway System) has a pattern matching and protocol anomaly approach with Detection Engine on the FPGA device as detection mechanism that can be applied to Gigabit-Ethernet links. Protocol anomaly detection is efficient detection mechanism to detect novel attacks.

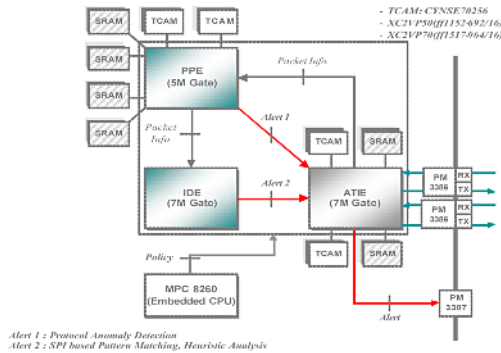
In this paper, we briefly introduce the whole architecture of our system designed to perform intrusion detection on high-speed links. And then, we present the protocol anomaly detection engine that pattern matching engine designed to perform intrusion detection on high-speed links. Finally, we conclude and suggest directions for further research.

## 2 System Architecture

We introduce the architecture of our system, named NGSS(Next Generation Security Systems) and components of the architecture.



**Fig. 1.** The Architecture of NGSS

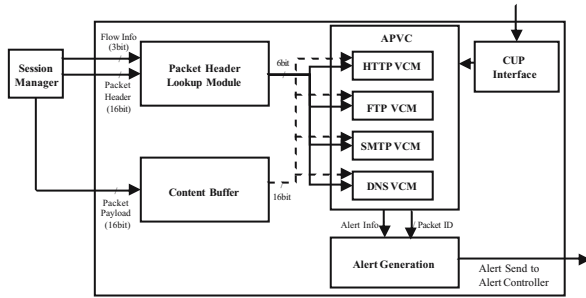


**Fig. 2.** Security Engine Board

NGSS consists of SGSs and SMSs. SMS(Security Management System) provides a detection and response policy to each SGSs in the same domain. SGS analyses incoming traffic and detects attacks in the incoming traffic. SGS is capable of managing these several boards according to network environments that it is applied. For detecting network intrusions more efficiently on high-speed links, SGS is composed of three FPGA Chips. As shown in the Fig. 2, one is ATIE FPGA Chip for wire-speed packet forwarding and blocking, another is PPE FPGA Chip for packet preprocessing and protocol anomaly detection, and the other is IDE FPGA Chip for high-performance intrusion detection.

### 3 Protocol Anomaly Detection Engine

Protocol anomaly detection is efficient detection mechanism at higher network speeds. Because the amount of comparison that needs to be performed is much smaller and much more static than signature-mechanism. This mechanism is also capable of detecting new and unknown attacks by distinguishing between a packet streams that breach



**Fig. 3.** Protocol Anomaly Engine

**Table 1.** The list of Service field inspected for Application protocol

Application Protocol	Inspection Fields
HTTP Client/Server	Length of URL field Invalid value in URL field Length of Chunk Binary Characters in field Invalid Request Format Accept-Language Field Length
SMTP Client/Server	Length of Command Line Length of Email Address Length of Reply Line Command Syntax Unsafe Command
FTP Client/Server	Command Syntax Length of Command Line Length of Pathname
DNS Client/Server	Length of Lable Length of DNS name Length of UDP/TCP Message Invalid OPCODE

acceptable application protocol usage rules and a legitimate packet streams. This engine analyzes only received packets after session established. Session manager send flow information consists of client/server direction, Session information. Fig. 3 is a block diagram of Protocol Anomaly Engine. Protocol Anomaly engine is combined with Session Manger. When a new packet arrived from session manager, it arrives concurrently with flow information. Packet data is passed to the engine through a 16-bit bus. The header information of each packet is compared with the predefined header rule in Packet Header Lookup Module. This Module checks whether there is unusual combinations of TCP flags, IP fragmentation, and unusual TCP options in packet header. If not find any unusual value in the header, the packet’s payload in content buffer is sent to APVC (Application Protocol Validation Checker) unit analyzed the header of application Protocol. Each VCM(Validation Check Module) extracts the value of service fields can give rise to buffer overflow attack in incoming packet and then calculates the length of value. If the

length limits the predefined threshold, alert is generated. The predefined size is rooted in RFCs and appropriate standards can be configured through CPU Interface. The flowing Table shows the list of service field is analyzed by VCM.

## 4 Pattern Matching Detection Engine

### 4.1 Header Lookup Mechanism

In design of our system, header lookup mechanism of IDE FPGA Chip is performed by flexible header combination lookup algorithm. This algorithm compares pre-defined header related rulesets with header information of incoming packets. If the incoming packet is matched with existing header patterns, 256bits match result is sent to payload matching logic and traffic volume based analysis logic. For this operation, this algorithm uses three memory maps; TCAM Lookup Map for each header field matching, Rule Combination Check Map for multiple header field matching, and Sequence Check Map for don't care filed matching.

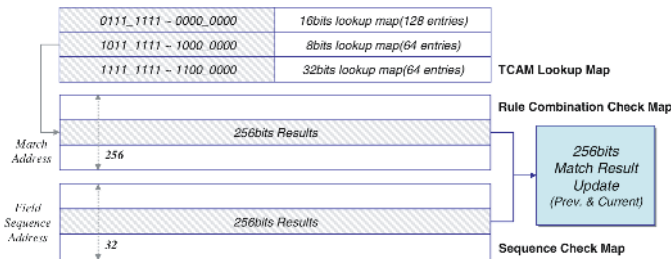


Fig. 4. Memory maps for packet header matching

### 4.2 Payload Matching Mechanism

In design of our system, payload matching mechanism of IDE FPGA Chip is performed by linked word based storeless running search algorithm. This algorithm compares pre-defined packet payload related rulesets with packet payload information of incoming packets. If the incoming packet is matched with existing payload patterns, alert message is generated according to the 256bits header lookup result. As shown in the above Fig. 5, reconstruction pattern length has boundary of size5 or 7 because of the limit of block memory in FPGA Chip.

Payload matching mechanism of IDE FPGA Chip is performed by linked word based storeless running search algorithm. This algorithm uses the spectrum dispersion technique to compare pre-defined packet payload related rulesets with packet payload information of incoming packets. As shown in the Fig. 6, the spectrum dispersion technique is method to calculate unique hash values about reconstructed patterns. For example, 5bytes "/etc/" pattern has the 9bits hash value "000001010" by sum about shifted values of each characters. These hash values are used as the rule memory address for each patterns.



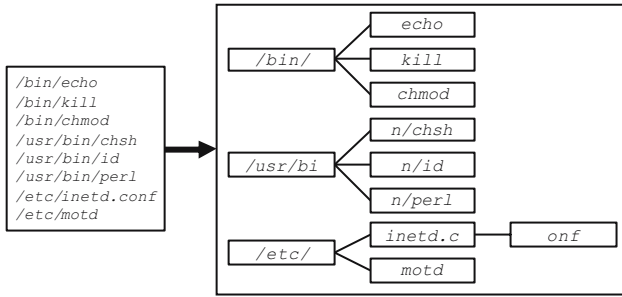


Fig. 5. Pattern Reconstruction

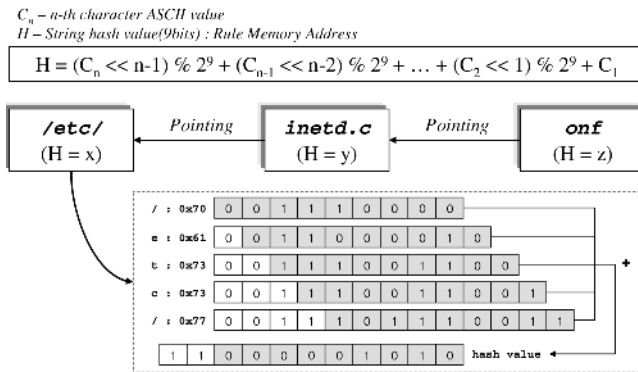


Fig. 6. Hash Value Calculation – Spectrum Dispersion Technique

After system booting, IDE FPGA Logic performs the hash value calculation about the incoming packet to the unit of byte. If the payload in incoming packet is matched with the pattern in memory pointed by the calculated hash value, then it is checked out which the related reconstructed patterns is matched or not. If all reconstructed patterns is matched with incoming packet, alert message is generated according to the header lookup results.

### 5 Conclusions

In this paper, we have treated two important problems in intrusion detection systems: Protocol Anomaly Detection and Pattern matching based Detection. We first presented a hardware based Protocol Anomaly Detection Engine to detect unknown attacks. The key idea of this is to apply the check module of Application Protocol Validation and the size of Service field to protocol anomaly engine to detect novel attacks in wire speed. The second issue we've handled in this paper is the problem of pattern matching mechanism in gigabit network. Our system is capable of processing until a maximum throughput of full-duplex 2Gbps about incoming packets in FPGA Logic.

Our final target for the development of SGS is to prevent all kinds of abnormal traffic from incoming the multi gigabit network without packet loss in real-time base. To achieve our goal, we will examine carefully the problem of our designed system by the process of test and make an more efforts to be a effectively solution.

## References

1. BoSong, Ming Ye, Jie Li: Intrusion Detection Technology Research based High-speed Network. IEEE PDCAT'2003 Proceedings (2003)
2. Enterasys Networks: Intrusion Detection Methodologies Demystified. (2003)
3. Byoung-Koo Kim, Jong-Su Jang, Sung-Won Sohn and Tai M. Chung: Design and Implementation of Intrusion Detection System base on Object-Oriented Modeling. In Proceedings of the International Conference on Security and Management (2002)
4. Kruegel, C., Valeur, F., Vigna, G. and Kemmerer, R.: Stateful intrusion detection for high-speed networks, In Proceedings of the IEEE Symposium on Security and Privacy. (2002)
5. M. Roesch: Snort-Lightweight Intrusion Detection for Networks, In Proceedings of the USENIX LISA '99 Conference. (1999)
6. Marcus Ranum: Burglar Alarms for Detecting Intrusions. NFR Inc. (1999)
7. S. Kumar and E. Spafford: A pattern matching model for misuse intrusion detection. In Proceedings of the 17th National Computer Security Conference. (1994)
8. W. Richard Stevens, TCP/IP Illustrated Volume I: The Protocols, Addison Wesley (1994)
9. Schuehler D.V, Moscola J, Lockwood J: Architecture for a hardware based, TCP/IP content scanning system, IEEE HOTI. (2003)
10. Byoung-Koo Kim, Ik-Kyun Kim, Ki-Young Kim, Jong-Soo Jang: Design and Implementation of High Performance Intrusion Detection System. ICCSA (2004)
11. Check Point Software Technologies: Multi-Layer Security: Attack Prevention Safeguards and Attacks Blocked, <http://cgi.us.checkpoint.com/securitycenter/whitepapers.asp>
12. Christoper Krugel, Thomas Toth, Engin Kirda: Service Specific Anomaly Detection for Network Intrusion Detection. In Symosium on Applied Computing(SAC). ACM. Scientific Press. (2002)

# Agent-Based Real Time Intrusion Detection System Against Malformed Packet Attacks\*

Jun-Cheol Jeon<sup>1</sup>, Eun-Yeung Choi<sup>2</sup>, and Kee-Young Yoo<sup>1, \*\*</sup>

<sup>1</sup> Department of Computer Engineering at Kyungpook National University  
Daegu, Korea, 702-701

`jcjeon33@infosec.knu.ac.kr`, `yook@knu.ac.kr`

<sup>2</sup> Planning and Administration Office at Seoul Metropolitan Office of Education  
Seoul, Korea, 110-781

`sionchoi@nate.com`

**Abstract.** The current paper proposes a network-based Intrusion Detection System (IDS) that can efficiently detect attacks based on malformed packets that continues to increase, along with more intelligent and skillful hacking techniques. Our system firstly extracts the important features from network packets and analyzes simple attacks and detects IP fragmentation attacks. Thereafter, it collects information from the SA and the FA and other strange information related to the malformed packet. Finally, it judges whether or not an intrusion has occurred on the basis of information gathered from target systems by CAs. The simulation result shows 0% false-positive and 0% false-negative, 100% detection ratio, thereby confirming the accuracy of the proposed IDS in detecting fragmentation attacks.

## 1 Introduction

Yet, hacking techniques have also become more intelligent and skillful, so that only one malformed packet can stop or even crash a network, and since most attacks are based on large-scale networks, for instance a LAN, WAN, or the Internet, the effects of such attacks are very serious. Denial of Service (DoS) attacks generally fall into two categories: stopping a service or resource exhaustion. Stopping a service means crashing or shutting off a specific server that users want to access, whereas, with resource exhaustion attacks, the service itself is still running, but the attacker consumes the computer network resources to prevent legitimate users from reaching the service [1]. Attacks based on malformed packets are particularly serious, as they cannot be properly detected by most IDSs mainly due to their various forms and sizes [2]. However, such attacks can be detected by analyzing the characteristics of the packets [3]. Accordingly, the current paper proposes a network-based IDS that can efficiently detect attacks by malformed packets in real-time.

---

\* This work was supported by the Brain Korea 21 Project in 2006.

\*\* Corresponding author.

## 2 Malformed Packet Attack

The following describes attacks based on malformed packets and hacking techniques that bypass the detection system.

**Simple Attack.** A DoS attack can damage and down a system, plus also stop a service and produce resource exhaustion. Although DoS attacks are usually achieved indiscriminately, a refined attack using just one malformed packet can crash a whole system [4], for examples, ping of death, jolt, teardrop, bonk, new teardrop and land attack.

**Bypassing Attack.** Most IDSs are based on a signature matching technique [6]. Therefore, some attackers avoid detection by forging packet data as if it is not a signature. Plus, attackers can also avoid detection through fragmentation, as IDSs do not provide a method for reassembling packets [7], for examples, tiny fragment attack and fragment overlap attack.

## 3 Proposed Intrusion Detection System

This section analyzes the characteristics of malformed packets outlined in previous research that can cause damage, then the proposed real-time IDS is illustrated to cope with such attacks.

### 3.1 Analysis of Malformed Packet Attacks

Based on the analysis of a monitored link, almost all packets in the link were found to be IP packets, with the great majority being TCP packets. UDP traffic comprised approximately 2% of the monitored traffic and IP packets that were neither TCP nor UDP made up an even smaller portion of the total data. Thus, the current analysis is based on the IP and TCP headers of the packets from the monitored traffic.

**Packet Header Size.** The IP header length should always be greater than or equal to the minimal Internet header length (20 octets) and a packet's total length should always be greater than its header length. If any of these statements do not hold for a given packet, it is invalid and should be discarded at the destination host. Yet, when an attacker forges the packet size to create a buffer overflow, forged packets can bypass an IDS due to insufficient information.

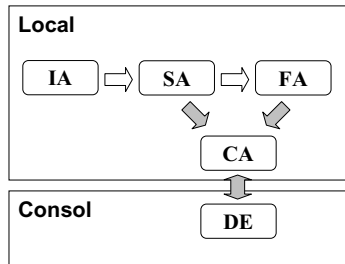
**Address and Port Number of Destination and Source.** Checking the source address and port number are important to detect a DoS attack or scanning attack. In most attacks, the source address or port number is forged, making it very difficult to find out the original source address of the attacker. Thus, if the IP address and port number (according to the service) of the source and destination are analyzed in detail, this could reduce the burden on the system. The following shows some check lists for intrusion detection based on spoofed addresses and port numbers.

- Inspect whether address for source and destination is the same.
- Inspect whether unknown service or port has been used to weaken security.
- Inspect whether specific host or network has been accessed.

**IP Fragment Filtering.** An IDS should be able to reassemble fragments to detect a bypassing attacks. Since the engine requires system resources, like memory, to process the function, a fragmentation attack is difficult to detect in real-time [7]. As such, the next section proposes an IP fragment-filtering algorithm for detecting a fragmentation attack in real-time.

### 3.2 System Configuration

The proposed system has five components as shown in Fig. 1: Information collecting Agent (IA), Simple analyzing Agent(SA), Fragment analyzing Agent (FA), Collaboration Agent (CA) and Decision Engine (DE).



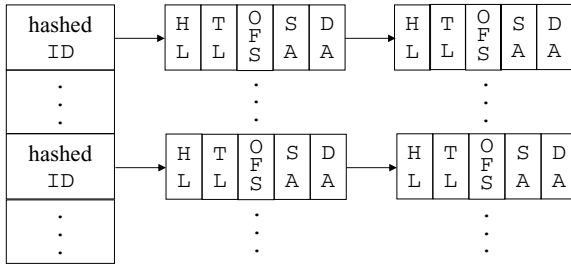
**Fig. 1.** System configuration: Information collecting Agent (IA), Simple analyzing Agent(SA), Fragment analyzing Agent (FA), Collaboration Agent (CA) and Decision Engine (DE)

**Information Collecting Agent.** The main task of the IA is to capture all packets in the Intranet. The IA then extracts the important features (parameters) from the network packets for use in the simple analyzer and fragment analyzer. To capture the packets, a BPF driver is used, as provided by Linux, which basically extracts the packet information related to intrusion using functions provided by a pcap-library [8]. The set of features,  $X$ , is in the form of 8-tuples of parameters, which are the main characteristics required to detect attacks using malformed packets. The following shows the set of features:

$$X=(TL, HL, DA, SA, DP, ID, FL, OFS)$$

where each parameter has the following meaning: (TL) total length of IP datagram in bytes, (HL) total length of datagram header in four-byte words, (DA) destination IP address of packet, (SA) source IP address of packet, (DP) destination port number, (ID) identification that shows datagram originated from source host, (FL) flags used in fragmentation, (OFS) fragmentation offset that shows relative position of fragment with respect to whole datagram.

The IA stores the features related to the packet fragmentation in a data structure to enable detection of a fragmentation attack at the FA. The features related to IP fragmentation must be inspected correctly to detect an IP fragmentation attack, otherwise bypassing attacks can occur. The IA stores the features related to IP fragmentation, then the FA analyzes them. As such, the ID, HL, TL, OFS, SA and DA are stored in a data structure. The ID is first hashed and then stored for performance and efficiency reasons, while the rest of the header information is stored at an adjacency linked list. Fig. 2 shows the data structure used to store the features.



**Fig. 2.** Data structure for storing features: ID is first hashed and then stored for performance and efficiency reasons, while the rest of the header information is stored at an adjacency linked list

**Simple Analyzing Agent.** The SA checks first whether the packet size is within a valid range, then second if the packet has the same destination IP address and source IP address. As such, an attack is detected by analyzing events with a set of detection rules. To identify intrusive patterns, based on the first five features of  $X$ , the sequence of packets must conform to the following conditions:

$$TL > 65,535 \text{ bytes} \wedge HL < 20 \text{ bytes} \wedge DA == SA$$

**Fragment Analyzing Agent.** Fragmented packets can arrive out of order, as they travel over different paths, yet, if one of the design assumptions is violated, undesired fragments can leak through the system. Fortunately, however, it is not necessary to remove all the fragments of an offending packet. Since "interesting" packet information is contained in headers, filters are generally only applied to the first and second fragments. The FA can also find the second fragment by comparing with the stored minimum offset. Fig. 3 describes the procedures of the IP fragment-filtering module. When the initial fragment (with a 0 offset) of an More Fragment (MF) flag arrives, the ID is hashed and stored in the structure. If a packet with a non-zeroed offset then comes, a check is made whether the second fragment has the same ID as the ID stored in the list. If the packet is the second fragment, the FA compares the total length of the packet. If the total length is larger than the offset, this means that the fragment is part of a fragment overlap attack. To prevent a buffer overflow, packets need to be deleted at a proper point.

```
IP fragment filtering module()
{
    IF flag is that MF an OFS is zero
        Store hashed ID, TL, SA, and DA
    ELSE
    THEN
    {
        Search for same ID in the structure
        If second fragment OFS is null
        THEN
            IF TL/8 > OFS
            THEN
                Alert and remove from buffer
            ELSE store minimal OFS
        ELSE second fragment OFS is not null
        THEN
            IF minimal OFS > present OFS
            THEN
                Alert and remove from buffer
        ELSE store minimal OFS
    }
}
```

**Fig. 3.** IP fragment-filtering module

**Collaboration Agent.** The CA has two functions. First, The CA collects result of analyzing at the SA and the FA then sends to the DE the information that gleans at each module. Secondly, The CA collects other strange information related to the malformed packet. The CA presents on each target system, migrates autonomously from host to host for exchanging information. Therefore attacks are detected more precisely.

**Decision Engine.** The DE is on the console machine and judges whether or not an intrusion has occurred on the basis of information gathered from target systems by CAs. The DE integrates information and evaluates it. All CAs bring information to the DE independently and, as a result, this information concentrated in the DE.

## 4 Simulation

The proposed system was implemented using a Linux Kernel 2.6 and 1400Mhz Intel pentium-4 PC, plus a pcap-library 0.6.2 was used to collect the packets and Teardrop and Nmap used as the attack tools. Attacks were attempted in various states to evaluate the performance of the proposed system. For a single-attack host, Teardrop and Nmap were used as a fragment overlap attack and a tiny fragment attack, respectively, plus we have attempted a fragment overlap attack and tiny fragment attack, together. Three multi-attack hosts have attempted each mentioned attack, and a mixed attack has also tried to investigate success or failure of intrusion detection on multi-attack hosts environments. In the case of a single-attack host, the simulation was performed thirty times, while in the case of multi-attack hosts, ten times of simulations were performed and a detection ratio calculated.

The detection results for the single-host attacks were 0% false-positive and 0% false-negative, 100% detection ratio. Also, the detection results for the multi-host attacks were the same, thereby confirming the effectiveness of the proposed system in detecting fragmentation attacks.

## 5 Conclusion

This paper proposed a network-based IDS that can efficiently detect malformed packets. First, patterns with malformed packet attacks are classified and the features related to the malformed packets are extracted. Next, these features are analyzed, while enables malformed packet attacks to be efficiently detected. In particular, the proposed IP fragment-filtering module can accurately detect a fragmentation attack.

## References

1. Ed Skoudis.: Counter Hack. Prentice Hall PTR (2002)
2. Paul E.proctor.: Practical Intrusion Detection Handbook. Prentice Hall PTR (2001)
3. Marina Bykova, Shawn Ostermann and Brett Tjaden.: Detection Network Intrusions via Statistical Analysis of Network Packet Characteristics. 33rd Southeastern Symposium on System Theory (SSST)(2001) 309–314.
4. Behrouz A.Forozan.: TCP/IP Protocol Suite. Mcgraw-Hill Companies, Inc (2000)
5. E. Biermann, E. Cloete and L.M Venter.: A comparison of Intrusion Detection System. Computers and Security, Vol. 20 (2001) 676–683
6. Stephen Northcut and Judy Novak.: Network Intrusion Detection An Analyst's Handbook Second Edition. New Riders (2001)
7. <http://www.cet.nau.edu/mc8/Socket/Tutorials/section4.html>.
8. Sang-Chul Kim.: Abnormal IP Packets. Korea Computer Emergency Response Team Coordination Center (2001)



# Efficient Mutual Authentication Scheme with Smart Card

Eun-Jun Yoon and Kee-Young Yoo\*

Department of Computer Engineering, Kyungpook National University,  
Daegu 702-701, South Korea  
Tel.: +82-53-950-5553; Fax: +82-53-957-4846  
ejyoon@infosec.knu.ac.kr, yook@knu.ac.kr

**Abstract.** In 2006, both Liu et al. and Yeh proposed an improvement of Chien et al.'s timestamp-based remote user authentication scheme with smart cards, that can withstand parallel session and forgery attacks, respectively. The current paper demonstrates that Liu et al.'s scheme is still vulnerable to a masquerading server attack and a more efficient and secure scheme is needed. One that not only resolves such problems, but also involves less computations and communication than these schemes.

**Keyword:** Network security, Secure protocol, Smart card, Authentication, Password.

## 1 Introduction

The remote password authentication scheme is a method that authenticates remote users over an insecure channel. As a protection mechanism of user authentication, the following criteria are crucial [4]: (1) User-independent server: No password or verification table is required to be kept in a server. (2) Freely-choose password: Whether users can choose their passwords freely. (3) Mutual authentication: Whether the users and the server can authenticate each other. (4) Lower communication and computation costs: The smart cards usually do not support powerful computation capability nor provide abundant bandwidth.

In 2002, Chien et al [1] proposed an efficient remote user authentication scheme, which satisfied all the above-mentioned criteria. In 2004, Hsu [2], however, pointed out that Chien et al.'s scheme is vulnerable to parallel session attack in which an adversary, without knowing a user's password, can masquerade as a legal user by eavesdropping or tampering with messages between the server and user. Thereafter, in 2005, Liu et al. [3] proposed an enhanced scheme. Liu et al. claimed that their scheme inherits all the merits of previous schemes as well as realizing secure mutual authentication without significantly increasing computational cost. In 2006, Yeh [4] also showed that Chien et al.'s scheme is insecure against a forgery attack because one adversary can easily pretend to

---

\* Corresponding author.

be a legal user, pass the server's verification and can login to the remote system successfully. Furthermore, Yeh proposed an improvement that can overcome security risks while still preserving the above advantages.

Nevertheless, the current paper will demonstrate that Liu et al.'s scheme [3] is vulnerable to a masquerading server attack, whereby an attacker can impersonate as a legal authentication server by creating a valid response message from the eavesdropped communication between authentication server and user. As a result, Liu et al.'s scheme fails to provide mutual authentication as the authors claimed. To remedy the attack, we present a more efficient and secure scheme in that it not only resolves such problems, but it also involves less computations and communication than both Yeh's [4] and Liu et al.'s schemes [3].

This paper is organized as follows: Section 2 briefly reviews both Yeh and Liu et al.'s timestamp-based remote user authentication scheme with smart cards. Section 3 discusses the weaknesses of Liu et al.'s scheme. The proposed scheme is presented in Sections 4, while Sections 5 discuss the security and efficiency of the scheme. Our conclusions are presented in Section 6.

## 2 Related Work

This section briefly reviews both Yeh's [4] and Lie et al.'s [3] timestamp-based remote user authentication scheme. These schemes can satisfy all criteria. We review these scheme in the following:

### 2.1 Yeh's Scheme

There are three phases in Yeh's scheme [4]: Registration, login, and authentication. The scheme works as follows:

**Registration Phase:** Let  $x$  be the secret key, which is the only secret maintained by the server, and  $h(\cdot)$  can be a secure one-way hash function with a fixed-length output. Assume that user  $U_i$  submits his identity  $ID_i$  and his password  $PW_i$  to the server for registration. The server computes  $R_i = h(ID_i \oplus x) \oplus PW_i$ , and issues a smart card which stores  $h(\cdot)$  and  $R_i$  to the user.

**Login Phase:** When a user  $U_i$  wants to login to the server, he attaches his smart card into a card reader. Then, he keys in his identity  $ID_i$  and password  $PW_i$ . Then, the following operations are performed by the smart card:

- (1) Compute  $C_1 = R_i \oplus PW_i$ .
- (2) Compute  $C_2 = h(C_1 \oplus T)$ , where  $T$  is the current time used as a timestamp.
- (3) Send the message  $(ID_i, T, C_2)$  to the server.

**Verification Phase:** After receiving the authentication request message  $(ID_i, T, C_2)$ , the server and the smart card execute the following jobs to facilitate mutual authentication between the user  $U_i$  and the server.

- (1) The server checks the validity of the  $ID_i$ , and verifies the time interval between  $T$  and  $T'$  in order to resist a replay attack, where  $T'$  is the timestamp when the request message is received.
- (2) The server computes  $C'_1 = h(ID_i \oplus x)$ , and verifies whether  $C_2? = h(C'_1 \oplus T)$ . If the verification fails, then the server rejects the request; otherwise, the server accepts  $U_i$ 's request and goes to Step (3).
- (3) The server acquires the current time stamp  $T''$ , and computes  $C_3 = h(C'_1 \oplus h(T''))$ . The server sends back the message  $(T'', C_3)$ .
- (4) Upon receiving the message  $(T'', C_3)$ ,  $U_i$  verifies the validity of the timestamp  $T''$ . Then,  $U_i$  verifies whether  $C_3? = h(C_1 \oplus h(T''))$ . If so,  $U_i$  believes that the responding part is the real server, and the mutual authentication process is completed; otherwise,  $U_i$  disconnects the connection.

## 2.2 Liu et al.'s Scheme

There are three phases in Liu et al.'s scheme [3]: Registration, login, and authentication. The scheme works as follows:

**Registration Phase:** Assume User  $U_i$  submits his identity  $ID_i$  and his password  $PW_i$  to the server for registration. The server computes  $R_i = h(ID_i \oplus x) \oplus h(PW_i)$ , and issues the smart card which stores  $h(\cdot)$ ,  $ID_s$ ,  $e$  and  $R_i$  for the user, where  $e$  is a random constant that is chosen by a server in the smart card and  $ID_s$  is the server's identity.

**Login Phase:** When a user  $U_i$  wants to login to the server, he attaches his smart card into a card reader. Then, he keys in his identity  $ID_i$  and password  $PW_i$ . Then, the following operations are performed by the smart card:

- (1) Compute  $C_1 = R_i \oplus h(PW_i)$ .
- (2) Compute  $C_2 = h(C_1 \oplus T)$ , where  $T$  is the current time used as a timestamp.
- (3) Compute  $M = h(e \oplus T)$ .
- (4) Send the message  $(ID_i, T, C_2, M)$  to the server.

**Verification Phase:** After receiving the authentication request message  $(ID_i, T, C_2, M)$ , the server and the smart card execute the following jobs to facilitate mutual authentication between the user  $U_i$  and the server.

- (1) The server checks the validity of  $ID_i$ , and verifies the time interval between  $T$  and  $T'$  in order to resist the replay attack, where  $T'$  is the timestamp when the request message is received.
- (2) The server computes  $C'_1 = h(ID_i \oplus x)$ , and verifies whether  $C_2? = h(C'_1 \oplus T)$ . If the verification fails, then the server rejects the request; otherwise, the server accepts  $U_i$ 's request and goes to Step (3).
- (3) The server computes  $M_s = h(ID_s \oplus M)$  and sends back the message  $M_s$ .
- (4) Upon receiving the message  $M_s$ ,  $U_i$  verifies whether  $M? = h(M_s \oplus ID_s)$ . If so,  $U_i$  believes that the responding part is the real server, and mutual authentication process is complete; otherwise,  $U_i$  disconnects the connection.

### 3 Cryptanalysis of Liu et al.'s Scheme

This section shows that Liu et al.'s scheme is vulnerable to a masquerading server attack. In the login phase, if an adversary has intercepted and blocked a message which was transmitted in Step (3), i.e.  $(ID_i, T, C_2, M)$ , he can impersonate a server and send  $M_s^* = h(ID_s \oplus M)$  to  $U_i$  in Step (3) of the verification phase, where  $ID_s$  is a server's public identity and  $M$  is the intercepted value. Upon receiving the first item of the received message, i.e.  $M_s^*$ ,  $U_i$  will compute  $h(M_s^* \oplus ID_s)$ . Note that Steps (1) and (2) of the verification phase are skipped by the adversary. Since the computed results equal the received message, i.e.  $M_s^*$ ,  $U_i$  will be fooled into believing that the adversary is the real server. Since  $U_i$  can not actually authenticate the server's identity, Liu et al.'s scheme fails to provide mutual authentication, as the authors claimed. The main key point, whereby the masquerading server attack can work is that all input values of  $M_s = h(ID_s \oplus M)$  are the publicly known values  $ID_s$  and  $M$ . Therefore, if an adversary intercepts the login message  $(ID_i, T, C_2, M)$ , he can create a valid response message  $M_s$  by using  $M$  and  $ID_s$ , and by passing the checking equations, where  $M^* = h(M_s \oplus ID_s)$ .

### 4 Improving Liu et al.'s Scheme

This section proposes an improvements to both Yeh's scheme and Liu et al.'s scheme, so that can withstand the security flaws described in previous sections, in order that they become more efficient. In order to overcome security risks and provide efficiency, we change the input values of  $M_s$ . We omit the registration and login phases due to that they are the same as Yeh's scheme and show it in the following.

**Verification Phase:** After receiving the authentication request message  $(ID_i, T, C_2)$ , the server and the smart card execute the following jobs to facilitate mutual authentication between the user  $U_i$  and the server.

- (1) The server checks the validity of  $ID_i$ , and verifies the time interval between  $T$  and  $T'$  in order to resist the replay attack, where  $T'$  is the timestamp when the request message is received.
- (2) The server computes  $C_1' = h(ID_i \oplus x)$ , and verifies whether  $C_2 = h(C_1' \oplus T)$ . If the verification fails, then the server rejects the request; otherwise, the server accepts  $U_i$ 's request and goes to Step (3).
- (3) The server computes  $C_3 = h(C_1' \oplus C_2)$ . The server sends back the message  $C_3$ .
- (4) Upon receiving the message  $C_3$ ,  $U_i$  verifies whether  $C_3 = h(C_1 \oplus C_2)$ . If so,  $U_i$  believes that the responding part is the real server, and mutual authentication process is complete; otherwise,  $U_i$  disconnects the connection.

### 5 Security and Efficiency Analysis

**Security Analysis:** The proposed scheme can resist a masquerading server attack. If a masquerading server tries to cheat the requesting user  $U_i$ , it has to

prepare a valid message  $C_3$ . This is infeasible, however, as there is no way to derive the value  $C_1 = h(ID_i \oplus x)$  in order to compute the value  $C_3 = h(C_1 \oplus C_2)$ . This is due to the one-way property of a secure one-way hash function [5]. Since we only modified Step (3) of the verification phase, other security requirements are satisfied as they are in Yeh's and Liu et al.'s schemes.

**Efficiency Analysis:** A comparison between Yeh's scheme [4], Liu et al.'s scheme [3] and our proposed scheme is shown in Table 1. In the login and verification phases, Yeh's scheme requires a total of seven hashing operations and six exclusive-or operations. Liu et al.'s scheme requires a total of seven hashing operations and seven exclusive-or operations. The proposed scheme, however, requires a total of five hashing operations and six exclusive-or operations. Furthermore, the user is only required to perform two hashing operations and three exclusive-or operations during the login and authentication phases of the proposed scheme. Obviously, the proposed scheme is more efficient than Yeh's and Liu et al.'s. Also, the proposed scheme uses a minimum communication bandwidth unlike both schemes. Among the four transmitted messages ( $ID_i, T, C_2, C_3$ ), one is the user's identifier (80 bit), one is a timestamp (80 bit) and two are hash output bits (160 bit such as SHA-1). These are very low communication messages.

**Table 1.** A comparison of computational costs

	Yeh's Scheme [4]		Liu et al.'s Scheme [3]		Proposed Scheme	
	User	Server	User	Server	User	Server
Login and Verification	3h+3xor	4h+3xor	4h+4xor	3h+3xor	2h+3xor	3h+3xor
Server Timestamp	Required		Not Required		Not Required	
Communication Costs	$\approx 560$ bits		$\approx 640$ bits		$\approx 480$ bits	

h: secure one-way hash operations; xor: bitwise exclusive-or operations.

## 6 Conclusions

The current paper demonstrated that Liu et al.'s scheme is vulnerable to a masquerading server attack and then, a more efficient and secure scheme was presented. The proposed scheme not only resolves such problems, but it also involves less computations and communication than both Yeh's and Liu et al.'s schemes.

## Acknowledgements

This research was supported by the MIC (Ministry of Information and Communication), Korea, under the ITRC (Information Technology Research Center) support program supervised by the IITA (Institute of Information Technology Assessment).

## References

1. Chien, H.Y., Jan, J.K., Tseng, Y.M.: An Efficient and Practical Solution to Remote Authentication: Smart Card. *Computers & Security*, Vol. 21. No. 4. (2003) 372-375
2. Hsu, C.L.: Security of Chien et al.'s Remote User Authentication Scheme Using Smart Cards. *Computer Standards & Interfaces*. Vol. 26. No. 3. (May 2004) 167-169
3. Liu, J., Sun, J., Li, T.: An Enhanced Remote Login Authentication with Smart Card. *The IEEE 2005 Workshop on Signal Processing Systems (SIPS'05)*. (November 2005) 229-232
4. Yeh, H.T.: Improvement of an Efficient and Practical Solution to Remote Authentication: Smart Card. *IEICE Transaction on Communication*. Vol. E89-B. No. 1. (January 2006) 210-211
5. Menezes, A.J., Oorschot, P.C., Vanstone, S.A.: *Handbook of Applied Cryptography*. CRC Press. New York. (1997)

# Strong Mobility for FIPA Compliant Multi-agent Systems

Javed Iqbal<sup>1</sup>, H. Farooq Ahmad<sup>2</sup>, Arshad Ali<sup>1</sup>, Hiroki Suguri<sup>2</sup>, and Sarmad Sadik<sup>1</sup>

<sup>1</sup>NUST Institute of Information Technology, Chaklala Scheme III, Rawalpindi, Pakistan  
+92-51-9280658

javed@piac.com.pk, {arshad.ali, sarmad}@niit.edu.pk

<sup>2</sup>Communication Technologies, 2-15-28 Omachi Aoba-ku Sendai, Japan  
+81-22-222-2591

{farooq, suguri}@comtec.co.jp

**Abstract.** In recent years popularity of Mobile-Agent systems makes it one of the promising technologies for developing intelligent software systems. The most important issues in the Mobile-Agent system are the migration of agent which can be categorized as Strong or weak. Strong mobility allows the agents to migrate without loss of execution state at any time, a powerful mechanism for implementing peer-to-peer computing environment. We have designed and implemented a strong mobility framework for SAGE (Scalable Fault Tolerant Agent Grooming Environment). Our key objective is to offer efficient and reliable infrastructure for agent's strong mobility. We have performed a number of experiments to quantitatively evaluate the effectiveness and efficiency of our proposed architecture.

## 1 Introduction

Mobile Agents is an active research area for the agent research community. It is changing the way distributed applications are developed and deployed [1]. It is still a developing research area, and a limited research progress has been made towards how to make mobility operations efficient and persistence by providing efficient mechanisms for persistence.

The motivation to make the agents mobile is load balancing, fault tolerance and dynamic reconfiguration of applications. Moreover, mobile agents can deal with non-continuous network connection, and as a consequence they suit mobile computing systems [4].

There are two types of mobility, weak and strong. In weak mobility only code and data state moves while in strong mobility in addition to code and data state, the execution state also moves along with agents. For distributed application Java is the first choice due to its universal portability and strong network programming. Java based mobility libraries can only provide weak mobility [5].

The reason to implement strong mobility for our system is that some application attributes like load balancing; resource sharing and fault tolerance cannot be achieved without capturing execution state of threads. We are using SAGE (Scalable Fault

Tolerant Agent Grooming Environment) [2] which is a FIPA[3] compliant multi agent system.

In the next sections we will present related work, system architecture, and in the last we have provided the performance measurements of our approach with conclusion and future directions.

## 2 Related Work

In order to capture the state of Java Thread two main approaches are followed: JVM-level approach and Application-level approach.

In JVM level approach researchers have introduced new functions to the Java environment to export the thread state from the JVM. In the Sumatra [6], Merpati [7], ITS [8] and CIA [9] projects, the JVM is extended with new mechanisms that capture a thread state in a serialized and portable form, but its main drawback is that it depends on a particular extension of the JVM; the provided thread serialization mechanism can therefore not be used on existing virtual machines.

In application level approach, the code is transformed to add new statements in the program. The added statements manage the thread state capture and restoration operations. Wasp [10] and JavaGo [11] provide a Java source code pre-processor while JavaGoX [12] rely on a bytecode pre-processor. The key advantage of application-level implementations is the portability for all Java environments.

Our instrumentation technique is different from others as our bytecode transformer can work automatically, also programmer can tell the transformer to checkpoint the code for optimization. In next section we are presenting our approach and Architecture.

## 3 Architecture for Strong Mobility

Agent in JVM executed as a cluster of Java objects that are working together to make the functionalities of the Agent. Each method call in JVM has its own call stack, which contains local variables, and operand stack. Single Java stack frame [13] is comprised of 3 parts .The operand stack, the execution environment and the local variable array. During process of Java serialization the information related to Java operand stack is lost can not be not serialized.

### 3.1 Transformer

For bytecode analysis and transformation purpose we used JOIE[14] bytecode transformer. A JOIE transformer may examine or modify any section of the class file. For example, it might add methods or insert instructions into an existing method, add new interfaces, or change the superclass. Transformer may add entries to the constant pool if, for example, it inserts instructions that reference new classes, fields, methods, or constants.



### 3.2 State Capture

As per FIPA specification [FIPA00023] and [PC00087A] an agent state at any point in time is represented by the flag (Boolean). For mobile agent FIPA has specified Active, Transit, Suspend and Resume states.

For execution state capture, every invoke instruction is analyzed and our transformation inserts code to check the FIPA State Flag of Agent. If state flag is set to Capture, in current method we store all information into SateObject Fig. 2 and return the control to the caller method and capture the related information, this process is repeated till the last caller method. In our approach user can put code to inform the transformer for checkpointing. This helped us to improve the optimization.

```
public class StateContext {
    public int[] ArrayI;
    public float[] ArrayF;
    public double[] ArrayD;
    public long[] ArrayL;
    public Object[] ArrayA;
    public Object[] ArrayThis
}
```

**Fig. 1.** Class For State Capture

In addition special values which include the state object for the current method, the state object for the caller of the current method.

### 3.3 State Restoration

Our transformer instrument byte code at start of each method and this inserted code is consists of several state reestablishing code blocks that checks the FIPA agent state and if it is Restore the code block then restores the method's frame to its original state and restores the method's pc register by jumping to the method's Program counter skipping the already executed instructions Figure-2 shows the pseudo code for restoration of State.

```
start of the method body
  Check if Resume status
  Check if this is the last stack frame then
    restoring = false;
    <get the program counter from myContext>
    <restore the stack frame>
    <go to the correct instruction>
```

**Fig. 2.** State Restoration Instruction

### 3.4 Migration

The sequence of steps involved in the migration of agent from source to destination is shown in figure 3. First of all Agent calls the method doMigration. The ACL message follows a FIPA-request protocol in which it specifies the destination where agent wants to migrate.

If the migration is accepted, the agent state is changed to transit, which means that agent is in transmission mode and state captures process starts and serialized object is then passed to network. Migration manager at destination loads the agent’s class, de-serializes its instance and restores its execution. For loading agent class we used our customized class loader, which had additional feature of loading classes from network.

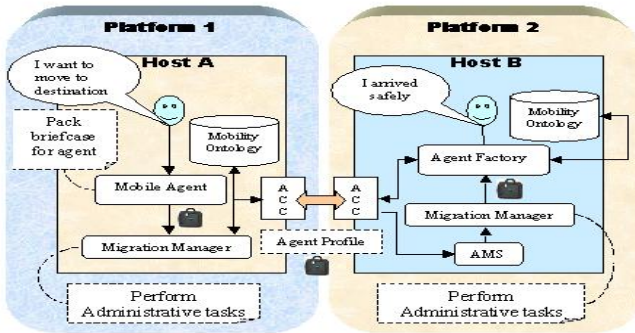


Fig. 3. Migration Process In Sage

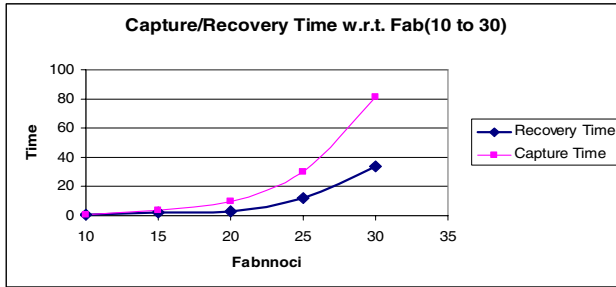
After successful operation, the agent state is changed to “execute” which means agent is now actively running in newly created thread.

## 4 Performance Evaluation

We performed several tests to measure the overhead imposed by our Implementation for capture, restoration and file space penalty with comparison to other approaches. We have chosen fabnnoci algorithm because of the great number of invocations that it performs. All tests were performed on a Pentium-IV overclocked to 1.6 GHz, 256 MB of RAM, running Windows XP SP-2.

### 4.1 Overheads Due to Capture and Restoration Operations

We calculated the overhead by using fabnnoci(30) series and result for capture and restoration are shown in graph 1 .

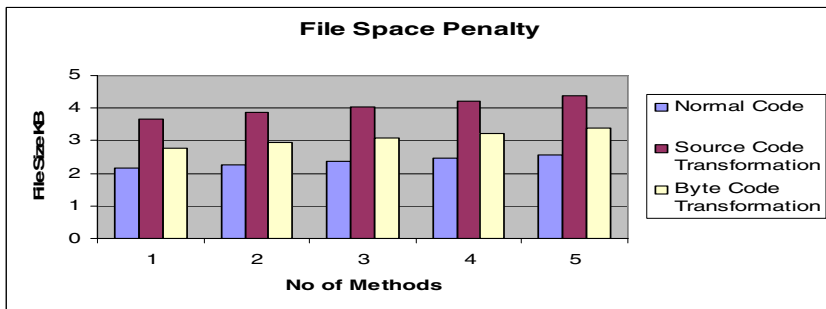


**Graph. 1.** Thread State Capture/Restoration

For similar code the overhead introduced by bytecode checkpointing in JavaGoX is 173ms, which shows that our system is efficient in terms of capturing and restoring state.

## 4.2 File Space Penalty

Bytecode instrumentation has some overheads in terms of file and space due to additional code.



**Graph. 2.** Byte Code Insertion with Respect to Original Code

Overall file space penalty calculated through various experiments for different nature of code is approximately 30%. Whereas if we use JavaGoX Bytecode implementation the file space penalty is approximately 183%.

## 5 Conclusions

We have presented a design and architecture for strong mobility for FIPA compliant multi agent systems. Our proposed architecture is fault tolerant, efficient and scalable. After carrying out various experiments, we confirmed that there are negligible overheads on system's performance due to transformation. Our proposed solution is

therefore feasible for both large and small systems. We have argued that strong mobility is an important abstraction for developing distributed applications. The API for the strongly mobile code and the translation mechanism are designed to give programmers full flexibility to choose between mobility types depending on nature of application. We are further working on generative strong migration to further reduce the overhead.

## References

1. A. Fuggetta, G. Picco, and G. Vigna, "Understanding Code Mobility", *IEEE Trans. Software Engineering*, May 1998, pp. 352-361.
2. Arshad Ali, H. Farooq Ahmad, Zaheer Abbas Khan, Abdul Ghafoor, Mujahid and Hiroki Suguri, "SAGE: Next Generation Multi-Agent System", in *Proceedings of the International Conference on Parallel and Distributed Processing Techniques and Applications*, Las Vegas, Nevada, USA 2004, pp. 139-145
3. Foundation for Intelligent & Physical Agents (FIPA). <http://www.fipa.org>, 2004.
4. Johansen, D., "Mobile Agent Applicability.", In, *Proceedings of the Mobile Agents 1998*, Springer-Verlag LNCS series Stuttgart, 9-11 September, 1998. Also in, *Journal of Personal Technologies*, Springer-Verlag, Vol 2, No. 2, 1999
5. Sun Microsystems Inc., "NFS: Network File System Protocol Specification", Tech. Report RFC 1094, file available for anonymous ftp from <ftp://nic.ddn.mil/directory/usr/pub/RFC>, 1989
6. Acharya A, Ranganathan M, Salz J. Sumatra: A language for resource-aware mobile programs. 2nd International Workshop 5 on Mobile Object Systems (MOS'96), Linz, Austria, July 1996. <http://www.cs.umd.edu/acha/publications.html>.
7. Suezawa T. Persistent execution state of a Java virtual machine. ACM Java Grande 2000 Conference, San Francisco, CA, June 2000. <http://www.ifi.unizh.ch/staff/suezawa/>.
8. Bouchenak S, Hagimont D. Pickling threads state in the Java system. Technology of Object-Oriented Languages and Systems Europe (TOOLS Europe'2000), Mont-Saint-Michel/Saint-Malo, France, June 2000.10
9. Illmann T, Krueger T, Kargl F, Weber M. Transparent migration of mobile agents using the Java platform debugger architecture. 5th IEEE International Conference on Mobile Agents (MA'2001), Atlanta, GA, December 2001.
10. Funfrocken S. Transparent migration of Java-based mobile agents (capturing and reestablishing the state of Java programs). 2nd International Workshop Mobile Agents 98 (MA'98), Stuttgart, Germany, September 1998.
11. Sekiguchi T, Masuhara H, Yonezawa A. A simple extension of Java language for controllable transparent migration and its portable implementation. 3rd International Conference on Coordination Models and Languages, Amsterdam, The Netherlands, April 1999. <http://liang.peng.free.fr/people-mobile.html>.
12. Sakamoto T, Sekiguchi T, Yonezawa A. Bytecode transformation for portable thread migration in Java. 4th International Symposium on Mobile Agents 2000 (MA'2000), Zurich, Switzerland, September 2000. <http://web.yl.is.s.u-tokyo.ac.jp/takas/>.
13. Lindholm, T., Yellin F., "The Java Virtual Machine Specification, Second Edition" Addison-Wesley, 1998.
14. Geo\_ A. Cohen, Jerrey S. Chase, and David L. Kaminsky. Automatic Program transformation with JOIE. In *USENIX 1998 Annual Technical Conference*, pages 167{178, June 1998.

# Author Index

- Ahmad, H. Farooq 714, 819  
Ahn, JinHo 754  
Ali, Arshad 714, 819  
Arai, Sachiyo 279
- Bae, Sang-Hyun 793  
Bai, Yang 269  
Bel-Enguix, Gemma 10  
Boshoff, W.H. 632
- Cao, Bing-gang 339  
Cao, Zining 46  
Chai, Yu-mei 432  
Chen, Hongbing 22  
Chen, Qingkui 571  
Chen, Shuang 455  
Chhabra, Manish 650  
Cho, Hyun-jin 420  
Choi, Eun-Yeung 807  
Choi, Jaeyoung 672  
Choi, Jonghwa 444, 502, 614  
Choi, Young-Keun 162
- Dignum, Frank 327  
Dong, Shoubin 150  
Dong, Zhiming 684  
Duo, Jiuting 684
- Ehlers, Elizabeth M. 34, 508, 632, 760  
Eom, Young Ik 420
- Ferreira, Chantelle S. 508
- Gao, Ji 94, 596  
Gao, Yuan 398, 478  
Gao, Zan 455  
Go, Christian Anthony L. 584  
Gong, Xun 187, 233  
Grando, María Adela 10  
Guan, Chun 534
- Hao, Jingbo 690  
He, Qing 129  
He, Qiu-sheng 696  
He, Yanxiang 71
- Heo, Jin-Kyoung 781  
Hu, Jinsong 546  
Hu, Jun 534  
Hu, Shan-Li 484, 490, 644  
Huang, Changqin 702  
Huang, He 199  
Huang, Hua-xin 94  
Huang, Tiyun 578  
Hur, SungMin 754
- Iqbal, Javed 819  
Ishida, Toru 1, 256, 293
- Jang, Jong-Soo 801  
Jee, Kyengwhan 773  
Jeon, Jun-Cheol 807  
Ji, Yuefeng 620  
Jia, Yan 386, 558, 564  
Jiang, Guorui 578  
Jiang, Wei 455  
Jiang, Weijin 608  
Jiang, Yichuan 256  
Jiménez-López, M. Dolores 10  
Jin, Beihong 463  
Jin, Li 596  
Jung, Gye-Dong 162
- Kang, Dong-Ho 801  
Ke, Youmin 490  
Kemke, Christel 84  
Kim, Byoung-Koo 801  
Kim, Gu Su 420  
Kim, Hyea Kyeong 678  
Kim, Il Kon 362  
Kim, Il Kwang 362  
Kim, Jae Kyeong 678  
Kim, Jong-Hun 708  
Kim, Jong-Min 626, 781  
Kim, Joonhyung 787  
Kim, Joung-Min 728  
Kimura, Mikako 734  
Kitamura, Yasuhiko 734  
Kook, Youn-Gyou 162  
Kwak, Byulsaim 59  
Kwon, Sungju 672

- Lam, Ka-man 138, 638  
 Lee, Dae-Young 793  
 Lee, Ho-Min 584  
 Lee, Jaeho 6, 59  
 Lee, Jae Young 362  
 Lee, Jung-Hyun 409, 708, 742  
 Lee, Juyeon 502  
 Lee, Kyoung Jun 678  
 Lee, Seungkeun 409, 742  
 Lee, Won-Hyung 584  
 Lee, Woong-Ki 626, 781  
 Leung, Ho-fung 138, 638  
 Li, Bin 339  
 Li, Hui 620  
 Li, Layuan 374  
 Li, Munan 546  
 Li, Rong 105  
 Li, Tao 187, 233  
 Li, Wei 463, 721  
 Li, Xiong 684  
 Li, Ya-chong 117  
 Liang, Gang 187  
 Liang, Xiao-hui 471  
 Liang, Zhengyou 150  
 Liao, Bei-shui 94, 596  
 Liao, Lejian 220  
 Liao, Rikun 620  
 Lim, Kee-Wook 708  
 Lin, Fen 199, 305, 534  
 Liu, Chunnian 664  
 Liu, Dongming 496  
 Liu, Kecheng 766  
 Liu, Peide 528  
 Liu, Xianggang 684  
 Liu, Xingquan 105  
 Liu, Zhiming 590  
 Lu, Hongen 650  
 Lu, Ji 187, 233  
 Lu, Zhaoxia 496  
 Luo, Jiewen 245, 305  
 Lv, Jianghua 175  
  
 Ma, Li 175  
 Ma, Shilong 175  
 Ma, Shu-gen 339  
 Ma, Yinglong 463  
  
 Na, Lichun 571  
 Nakanishi, Hideyuki 293  
 Nam, Taek-Young 801  
  
 Ni, Guoqiang 520  
 Nie, Lanshun 315  
 Nie, Zefeng 520  
 Nisikata, Takumi 656  
  
 Oh, Jin-Tae 801  
 Oosthuizen, Ockmer L. 760  
 O'Reilly, Grant Blaise 34  
 Osman, Mashanum 748  
  
 Pan, Jing 175  
 Park, Seung-Kyu 728, 781  
 Pasha, Maruf 714  
 Pasquier, Philippe 327  
 Peng, Hong 546  
 Peng, Yuxing 590  
  
 Qiu, Jing 220  
 Qiu, Lirong 199  
  
 Rahwan, Iyad 327  
 Razali, Szalinsyah 748  
 Rong, Wenge 766  
  
 Sadik, Sarmad 819  
 Sawamura, Hajime 656  
 Shi, Chuan 305  
 Shi, Chun-Yi 484  
 Shi, Man-Yin 644  
 Shi, Zhongzhi 5, 199  
 Shin, Dongil 444, 502, 614, 787  
 Shin, Dongkyoo 444, 502, 614, 787  
 Shui, Chao 386, 558  
 Son, Minwoo 787  
 Sonenberg, Liz 327  
 Song, Chang-Woo 708  
 Song, Mang-Kyu 781  
 Suguri, Hiroki 714, 819  
 Sun, Feixian 187  
 Sun, Yu 150  
  
 Tan, Yu-An 351  
 Tanaka, Nobuyuki 279  
 Tanaka, Rie 293  
 Tang, Ming Xi 514  
 Tao, Junwei 455  
 Tao, Yang 590  
 Tu, Shi-liang 696  
  
 Wang, Chao 455  
 Wang, CuiRong 398, 478  
 Wang, Huaiming 386

- Wang, Hui 398, 478  
 Wang, Jian Xun 514  
 Wang, Jie 664  
 Wang, Jun 564  
 Wang, Kai 684  
 Wang, Lianlai 552  
 Wang, Li-ming 117, 269  
 Wang, Maoguang 245, 534  
 Wang, Shu-Wu 351  
 Wang, Tiefang 187  
 Wang, Wei 211  
 Wang, Zheng-guang 471  
 Wang, Zhong-feng 432  
 Wu, Jinghua 578  
 Wu, Kehe 463  
 Wu, Quan-Yuan 564  
 Wu, Zhaohui 7  
  
 Xiao, Jun 71  
 Xing, Li-Ning 351  
 Xu, Fuyin 702  
 Xu, Manwu 22  
 Xu, Tingfa 520  
 Xu, Wen-bo 602  
 Xu, Xianghua 702  
 Xu, Xiaofei 315  
 Xu, Yusheng 608  
 Xue, Jinyun 552  
 Xue, Xiao 105  
  
 Yan, Xin 374  
 Yang, DeGuo 398, 478  
 Yang, Gongping 496  
 Yang, Hwan-Seok 626, 728, 781  
 Yang, Jin 187  
  
 Yang, Jung-Jin 773  
 Yang, Qun 22  
 Yao, Jianmin 520  
 Yin, Jianping 690  
 Yoo, Kee-Young 807, 813  
 Yoon, Eun-Jun 813  
 Yu, Qing 199, 245  
 Yuan, Lulai 211  
  
 Zeng, Guangzhou 496  
 Zeng, Guosun 211  
 Zeng, Li 245  
 Zhan, Dechen 315  
 Zhang, Boyun 690  
 Zhang, Huaxiang 528  
 Zhang, Li-ping 339  
 Zhang, Ling 150  
 Zhang, Sulan 129  
 Zhang, Xi-huang 602  
 Zhang, Xin-Hua 351  
 Zhang, Xuejie 540  
 Zhang, Xue-Lan 351  
 Zhang, Zehua 540  
 Zhang, Zheng 339  
 Zhao, Qin-ping 471  
 Zhao, Xiurong 129  
 Zhao, Yuhui 478  
 Zhao, Zhikun 721  
 Zheng, Di 564  
 Zheng, Xiaolin 702  
 Zheng, Yujun 552  
 Zhou, Bing 558  
 Zhou, Pen 386, 558  
 Zhu, Liehuang 220